

# **Artificial Intelligence Approach to Damage Detection in Lightweight Structures**

**André Guilherme Massano Tavares**

Thesis to obtain the Master of Science Degree in

## **Aerospace Engineering**

Supervisors: Prof. Nuno Miguel Rosa Pereira Silvestre  
Prof. Giuliano Coppotelli

### **Examination Committee**

Chairperson: Prof. Fernando José Parracho Lau  
Supervisor: Prof. Nuno Miguel Rosa Pereira Silvestre  
Member of the Committee: Prof. Pedro Manuel Ponces Rodrigues de Castro Camanho

**July 2020**



Dedicado ao meu irmão



## Acknowledgments

The work developed in this project represents the application of the accumulative knowledge and work method developed across many years of university study, achieved with the help of many people, to whom I would like to express my gratitude.

First of all, to Instituto Superior Técnico of Lisbon, the faculty that saw me grow as a person and a future engineer, since 2014. I would like to address some professors that every day contribute to maintaining and enhancing the quality of teaching from one of the most prestigious faculties in Portugal, who marked my path with their personalities and teachings. Moreover, I would like to make a special reference to Professor Nuno Silvestre, who was my supervisor and awoke my curiosity to the field of structural mechanics, with his Master's course.

Secondly, to Facoltà di Ingegneria Civile e Industriale of the Università La Sapienza di Roma, the faculty of my Master's double degree, where I spent one very marking year of my life. I feel grateful to have been well hosted by the professors, and to have had this opportunity also to be in contact with the Italian culture. I would like to make a special reference to Professore Giuliano Coppotelli, to whom I feel grateful for having been his Master's thesis student and for giving me such an important opportunity to make an internship at the Siemens Industry Software NV company.

To Siemens Industry Software NV, I would like to express my gratitude to my supervisor, Eng. Emilio di Lorenzo, who provided for a great internship experience, by guiding my work from the start always showing his availability to help me along the way. Moreover, to all the new friends I met in the Test Division office, from three different continents, with whom I had the most interesting conversations.

To all the friends I have made during my university course, in the last 5 years, both in Lisbon and in Rome, I would like to leave a special acknowledgement. Many adventures, stories will remain always in my heart to remember a ride of excitement, hard work and friendship. Among many others, I would like to acknowledge Emanuel Canelas, David Alexandre, Rodrigo Oliveira, Rita Moreira, Eunice Oliveira, Teresa Araújo.

To all my friends I bring from my hometown, I feel so grateful for the most amazing friendships, adventures, experiences and learning. With them, I learned the value of friendship, and wherever I go, in the travels I make, I always carry these people in my thought and heart. To André Basílio, André Cabral, Fernando Nunes, João Bernardo, João Marques, José Alves, Luís Ferreira, Marta Riquito, Miguel Nunes, Rodrigo Abreu, Rui Ferreira, Salomé Moreira, I thank for all the coffees, vacations, trips, and calm dinners.

To my grandparents, from whom, during my life, I heard stories of perseverance, love, intelligence, dedication, they keep inspiring my way of being.

To my parents, Paulo and Carla, for all the unconditional support and love they gave, I owe everything and feel thankful for the education they gave me, which brought me to where I am. I remain ambitious and eager to keep sharing my life and news with them, as a son. To my brother Tiago, I dedicate the work I have done, for he has a special place in my heart and with him I grew up, getting inspired by his relentless spirit.



## Resumo

Uma fidedigna e eficaz detecção de dano é crítica para o uso de materiais leves na indústria mecânica e aeroespacial. No contexto de Teste Não-Destrutivo (TND), testes de vibração têm sido aplicados por muitas décadas para inspecionar componentes sem danificar ou debilitar o seu uso. Para posterior detecção de dano, técnicas de Inteligência Artificial (IA) alcançaram grande sucesso em várias aplicações estruturais.

Neste trabalho, Teste, Simulação e Inteligência Artificial (IA) foram combinados para desenvolver procedimentos de detecção de dano. O uso de um Optomet Scanning Laser Doppler Vibrometer (SLDV) para estes testes providencia uma solução interessante para medir a velocidade de vibração na superfície de uma estrutura. Os algoritmos para identificar defeitos são baseados no princípio de Resonância Local de Dano (RLD), que olha para as altas frequências de vibração de modo a obter a activação localizada da resonância do defeito.

Técnicas de Inteligência Artificial (IA) foram implementadas com o objetivo de criar um procedimento automático combinado com extração de parâmetros para detecção de dano. A transformação de Wavelet e análise modal foram usadas para providenciar inputs para as técnicas de IA.

De modo a perceber melhor as limitações em termos de detecção de dano, placas defeituosas foram moduladas e simuladas de modo a criar uma análise de sensibilidade. Em último, uma comparação geral dos diferentes algoritmos foi obtida.

**Palavras-chave:** Placas leves, TND, Laser Doppler Vibrometer, detecção de dano, IA





## Abstract

Reliable and efficient damage detection is critical for the use of lightweight materials in the mechanical and aerospace industries. Within the context of Non-Destructive Testing (NDT), vibration-based tests have been applied for many decades to inspect components without damaging or debilitating their use. For posterior fault recognition, Artificial Intelligence techniques have achieved high success for a number of structural applications.

In this work, Testing, Simulation and Artificial Intelligence have been combined in order to develop a defect detection procedure. The use of an Optomet Scanning Laser Doppler Vibrometer (SLDV) for such tests provides an interesting solution to measure the vibration velocities on the structure surface. The algorithms for identifying the defects are based on the Local Defect Resonance (LDR) concept, which looks to the high frequency vibrations to get a localized resonant activation of the defect.

Artificial Intelligence (AI) techniques were implemented with the aim of creating an automatic procedure combined with feature extraction for damage detection. Wavelet transformation and modal analysis were used to provide inputs to the AI techniques.

In order to better understand the limitation in terms of defect detection, damaged plates were modelled and simulated in order to perform a sensitivity analysis. Finally, an overall comparative overview of different algorithms results was also obtained.

**Keywords:** Lightweight plates, NDT, Laser Doppler Vibrometry, damage detection, AI



# Contents

Acknowledgments . . . . .	v
Resumo . . . . .	vii
Abstract . . . . .	ix
List of Tables . . . . .	xv
List of Figures . . . . .	xvii
<b>1 Introduction</b>	<b>1</b>
1.1 Topic Overview . . . . .	1
1.2 Motivation . . . . .	3
1.3 Objectives . . . . .	5
1.4 Thesis Outline . . . . .	5
<b>I Theoretical Background</b>	<b>6</b>
<b>2 Local Defect Resonance</b>	<b>7</b>
<b>3 Machine Learning</b>	<b>8</b>
3.1 Basic concepts . . . . .	8
3.2 K-means Clustering . . . . .	10
3.2.1 Algorithm . . . . .	10
3.2.2 Visualization . . . . .	11
3.3 Anomaly Detection based on Multivariate Gaussian Distribution . . . . .	13
3.3.1 Algorithm . . . . .	13
3.3.2 Visualization . . . . .	19
<b>4 Deep Learning</b>	<b>20</b>
4.1 Convolutional Neural Networks . . . . .	20
4.1.1 1D vs 2D convolutional neural networks . . . . .	22
4.2 Transfer Learning . . . . .	22
<b>5 Bayesian Optimization</b>	<b>24</b>

<b>II</b>	<b>Defect Detection for Experimental Data</b>	<b>25</b>
<b>6</b>	<b>Experimental Setup</b>	<b>27</b>
<b>7</b>	<b>Time Data Procedure</b>	<b>29</b>
7.1	Time Data . . . . .	29
7.1.1	Time Signals . . . . .	30
7.1.2	CWT and STFT Images . . . . .	30
7.1.3	CWT and STFT Maximum Amplitudes . . . . .	33
7.2	Machine Learning Tool (K-means) . . . . .	33
7.2.1	Methodology . . . . .	34
7.2.2	Results . . . . .	34
7.3	2D CNN (Convolutional Neural Network) . . . . .	36
7.3.1	Methodology . . . . .	36
7.3.2	Results . . . . .	37
7.4	1D CNN (Convolutional Neural Network) . . . . .	39
7.4.1	Methodology . . . . .	39
7.4.2	Results . . . . .	40
7.4.3	Bayesian Optimization . . . . .	41
<b>8</b>	<b>Frequency Data Procedure</b>	<b>44</b>
8.1	Frequency Data . . . . .	44
8.1.1	Frequency Response Functions (FRFs) . . . . .	45
8.1.2	Mode shapes . . . . .	45
8.1.3	2nd Gradients of mode shapes . . . . .	46
8.2	Machine Learning Tool (GADAC) . . . . .	47
8.2.1	Methodology . . . . .	48
8.2.2	Results for the mode shapes . . . . .	49
8.2.3	Results for the 2nd Gradients of the mode shapes . . . . .	50
8.3	1D CNN (Convolutional Neural Network) . . . . .	52
8.3.1	Methodology . . . . .	52
8.3.2	Results . . . . .	53
8.3.3	Bayesian Optimization . . . . .	54
<b>III</b>	<b>Defect Detection for Simulation Data</b>	<b>56</b>
<b>9</b>	<b>Simulation Setup</b>	<b>58</b>
9.1	Part Model . . . . .	58
9.2	Finite Element Model . . . . .	59
9.3	Simulation Model . . . . .	59

<b>10 SOL 103 - Real Eigenvalues Procedure</b>	<b>61</b>
10.1 Simulation data . . . . .	62
10.1.1 Mode shapes . . . . .	62
10.1.2 2nd Gradients of mode shapes . . . . .	63
10.2 Machine Learning Tool (K-means) . . . . .	64
10.2.1 Pre-processing . . . . .	64
10.2.2 Methodology . . . . .	65
10.2.3 Results for the mode shapes . . . . .	65
10.2.4 Results for the 2nd Gradients of the mode shapes . . . . .	66
10.2.5 Best version of the 2nd Gradient algorithms . . . . .	67
10.3 Robustness analysis . . . . .	69
<b>11 Other results</b>	<b>73</b>
11.1 CFRP plate 1 . . . . .	73
11.2 PMMA plate . . . . .	75
11.3 CFRP plate 2 . . . . .	76
<b>Conclusion and Future Developments</b>	<b>77</b>
Match of all techniques . . . . .	78
Future developments . . . . .	79
<b>Bibliography</b>	<b>81</b>
<b>A 1D CNN architectures</b>	<b>85</b>
A.1 First 1D CNN . . . . .	85
A.2 1D CNNs obtained with the Bayesian Optimization . . . . .	86
A.2.1 For Time-domain data . . . . .	86
A.2.2 For Frequency-domain data . . . . .	87
<b>B Machine Learning tools applied to singular mode shapes and 2nd Gradients</b>	<b>88</b>
B.1 Experimental mode shapes . . . . .	88
B.2 Simulated mode shapes . . . . .	89



# List of Tables

3.1	Influence of covariance matrix's equal diagonal values . . . . .	14
3.2	Influence of covariance matrix's different diagonal values and the mean vector's values . .	15
3.3	Influence of covariance matrix's non diagonal values and the mean vector's values . . . .	16
3.4	Confusion matrix . . . . .	18
7.1	Parameters left for Bayesian Optimization regarding the training and architecture of the 1D CNN . . . . .	42
7.2	Optimized training hyperparameters for the time-domain 1D CNN . . . . .	42
8.1	Optimized training hyperparameters for the frequency-domain 1D CNN . . . . .	54
9.1	2D part model properties . . . . .	59
9.2	2D finite element model properties . . . . .	59
9.3	Simulation model properties . . . . .	60
10.1	Results for the modelled plate with FBH thickness = 30 % . . . . .	68
10.2	Robustness analysis results . . . . .	71
11.1	Match of all results from the different algorithms for the experimental procedure . . . . .	78
A.1	1D CNN architecture used for the time and frequency-domain analysis . . . . .	85
A.2	Optimized architecture for the time-domain 1D CNN . . . . .	86
A.3	Optimized architecture for the frequency-domain 1D CNN . . . . .	87





# List of Figures

1.1	Aircraft development process [26]. . . . .	4
3.1	Machine learning algorithm process. . . . .	9
3.2	Representation of a a) supervised learning and b) unsupervised learning algorithms. . . . .	10
3.3	K-means Clustering progression. . . . .	12
3.4	K-means limitations to clustering non-spherical datasets. . . . .	12
3.5	Multivariate Gaussian vs Gaussian distribution threshold curves. . . . .	14
3.6	Anomaly Detection progression. . . . .	19
4.1	Deep Learning and Machine Learning panorama inside the Artificial Intelligence area. . . . .	20
4.2	Sigmoid and ReLU activation functions. . . . .	21
4.3	Convolutional neural network architecture representation with the usual convolutional operations before the fully connected layers. . . . .	22
4.4	One and two-dimensional convolutional and pooling operations comparison. . . . .	23
4.5	Inception-ResNet-v2 architecture. . . . .	23
5.1	Algorithm methodology developed in this thesis project for experimental data. . . . .	26
6.1	Experimental setup with the LDV and the plate to be measured below over a piece of foam. To the right is presented an example of a measured grid by the LDV . . . . .	27
6.2	PMMA plate, from whose measurements will be used to present the algorithms' results. . . . .	28
7.1	Overview of the methodologies developed for the time-domain analysis a) Time data procedure and b) input data for each of the corresponding algorithms. . . . .	29
7.2	Time signals measured with the LDV on the PMMA plate. . . . .	30
7.3	Continuous Wavelet Transform (CWT) applied to 4 different <b>damaged</b> time signals. . . . .	31
7.4	Continuous Wavelet Transform (CWT) applied to 4 different <b>non-damaged</b> time signals. . . . .	31
7.5	Short-Time Fourier Transform (STFT) applied to 4 different <b>damaged</b> time signals. . . . .	32
7.6	Short-Time Fourier Transform (STFT) applied to 4 different <b>non-damaged</b> time signals. . . . .	32
7.7	Result of the application of the CWT and STFT to the time-signals so to obtain the maximum vibration amplitudes. A vector of points like the one displayed in a) is obtained. Figures b) and c) show the plot of the respective CWT and STFT vector on a 2D grid with the dimension of the grid measured by the LDV. . . . .	33

7.8 Measured PMMA plate correspondent to the results to be presented in part II (Defect Detection for Experimental Data). . . . .	35
7.9 ML tool (K-means) applied on the CWT maximum amplitudes result. . . . .	35
7.10 ML tool (K-means) applied on the STFT maximum amplitudes result. . . . .	35
7.11 GoogLeNet architecture [14]. . . . .	37
7.12 Label map for the CNNs . . . . .	37
7.13 2D CNN applied on the CWT time-frequency image representation result. . . . .	38
7.14 2D CNN applied on the STFT time-frequency image representation result. . . . .	38
7.15 Overlap operation. . . . .	40
7.16 1D CNN for time-domain data result. . . . .	41
7.17 Optimization methodology representation. . . . .	42
7.18 Figure a) represents the results obtained with the optimized 1D CNN on the PMMA plate, which was obtained from the Bayesian Optimization in relation to the labels in figure b). . . . .	43
8.1 Overview of the methodologies developed for the frequency-domain analysis a) Frequency data procedure and b) input data for each of the corresponding algorithms. . . . .	44
8.2 FRFs measured with the LDV on the PMMA plate. . . . .	45
8.3 (a) Mode shape absolute vector. (b) Plot of the absolute of the mode shape vector into the LDV measured grid . . . . .	46
8.4 2nd gradient procedures. . . . .	46
8.5 Comparison of a) reference mode shape and the result of applying the b) Laplacian and c) Gaussian Curvature 2nd Gradient procedures. . . . .	47
8.6 Plot of the 2nd Derivatives amplitudes in relation to the base mode shape ( $\phi$ ) in Figure 8.5a	47
8.7 Anomaly detection based in Multivariate Gaussian Distribution representation. . . . .	49
8.8 ML tool implementation methodology related to the Polymax mode shape calculation. . . . .	49
8.9 ML tool (GADAC) results for the mode shape data. . . . .	50
8.10 ML tool implementation methodology related to the Polymax mode shape calculation and the 2nd Gradient application. . . . .	51
8.11 ML tool (GADAC) - perspective view of the total defect maps obtained for each of the 2nd Gradients. . . . .	51
8.12 ML tool (GADAC) - top view of the total defect maps obtained for each of the 2nd Gradients.	51
8.13 1D CNN for frequency-domain data result. . . . .	54
8.14 Figure a) represents the results obtained with the optimized 1D CNN on the PMMA plate, which was obtained from the Bayesian Optimization in relation to the labels in figure b). . . . .	55
8.15 Algorithm methodology developed in this thesis project for simulation data. . . . .	57
9.1 Overall simulation procedure. . . . .	58
9.2 Match between the modelled and the real plates. Both have similar Flat Bottom Holes sizes and shapes . . . . .	59
9.3 Finite Element Analysis. . . . .	60

9.4	Examples of obtained modes. . . . .	60
10.1	Overview of the simulation data and methodologies used for the 2nd Gradient techniques. . . . .	61
10.2	Difference between a regular grid of equidistant points as the one performed by the LDV , and a grid of points obtained by the finite elements' DOFs. . . . .	62
10.3	Comparison between the irregular simulation and the regular interpolation grids. . . . .	63
10.4	Plot of the Laplacian and Gaussian Curvature amplitudes with reference to the mode shape interpolation into a regular grid, presented in figure 10.3b. . . . .	63
10.5	Plot of the 2nd Derivatives amplitudes with reference to the mode shape ( $\phi$ ) interpolation into a regular grid, presented in figure 10.3b. . . . .	64
10.6	Pre-processing context inside both Machine Learning tools procedure. . . . .	65
10.7	Visualization of the clustering process . . . . .	66
10.8	Results for a modelled plate with 30% thickness and a mesh size of 2 mm. . . . .	66
10.9	Detection rate results of the Machine Learning tool applied to the 2nd Gradients of the mode shapes calculated for a modelled plate with 30% thickness and a mesh size of 2 mm. . . . .	67
10.10	Top view of the detection rate results of the Machine Learning tool applied to the 2nd Gradients of the mode shapes calculated for a modelled plate with 30% thickness and a mesh size of 2 mm. . . . .	67
10.11	Robustness analysis methodology. . . . .	69
10.12	Maximum detection rate values for the ML tool applied to a) mode shapes and b) 2nd Derivatives of the mode shapes. . . . .	70
11.1	1st CFRP plate's distribution of FBHs and a representation of the measured grid by the LDV (dashed red rectangle). . . . .	73
11.2	CFRP plate 1 - ML tool (GADAC) results . . . . .	74
11.3	CFRP plate 1 - 2D CNN results . . . . .	74
11.4	CFRP plate 1 - 1D CNN results . . . . .	75
11.5	PMMA plate's distribution of FBHs and a representation of the measured grid by the LDV (blue rectangle). . . . .	75
11.6	PMMA plate (4 defects) - ML tool (GADAC) results . . . . .	75
11.7	PMMA plate (4 defects) - 1D CNN results . . . . .	76
11.8	2nd CFRP plate's distribution of FBHs and a representation of the measured grid by the LDV (red dashed rectangle). . . . .	76
11.9	CFRP plate 2 - ML tool (GADAC) results . . . . .	76
B.1	PMMA plate and corresponding high-frequency mode shape. . . . .	88
B.2	Results of the ML tool (GADAC) applied on a single mode shape (figure B.1b) and its 2nd Gradients . . . . .	89
B.3	Simulated plate and corresponding high-frequency mode shape. . . . .	89

B.4 Results of the ML tool (K-means) applied on a single mode shape (figure 8.5a) and its  
2nd Gradients . . . . . 90

# 1

## Introduction

### 1.1 Topic Overview

Early detection of damages in production lines could prevent serious economical dispenses and also major device malfunctions. Reliable and efficient damage detection is critical for the utilization of composite materials and a great challenge in the mechanical, aerospace and aeronautical industries. Therefore a solid defect detection methodology will guarantee the integrity and security of structures along with reducing the maintenance and repair costs. Common methods of visual inspection are time consuming and not effective detecting damages that are not manifested on the surface, like inserts. Thus, methods of Non-Destructive Testing (NDT) are used.

Composite materials are composed of two or more distinct phases (matrix and fiber), whose overall properties are superior to those of the individual components. Thus achieving high strength to weight ratio, high tensile strength at elevated temperatures having good fatigue resistance associated with a weight reduction compared to the individual components [1] [2]. With these unique properties, these materials have been highly used throughout different industries like the automotive, aerospace and to build different complex structures such as fuselages, wind turbine blades, etc.

#### **Structural Health Monitoring context with Local Defect Resonance**

Within this context, Structural Health Monitoring (SHM) and vibration-based tests have been applied on the mechanical and aeronautical industries for many decades. In terms of definition, SHM can be defined as "acquisition, validation and analysis of technical data to facilitate life-cycle management of decisions" [3]. But more generally, it denotes a reliable system with the ability to detect and interpret adverse changes in a structure due to damage or normal operation [4]. In the context of damage detection, these techniques benefit largely from the concept of Local Defect Resonance (LDR).

LDR makes use of high frequency vibrations to get localized resonant activation of the defect, since it leads to a decrease in stiffness for a certain mass of the material in that area. Using uncommonly high excitation frequencies, this technique represents an extension of traditional modal analysis frequency regimes, but associated with it comes the localized defect activation providing a high contrast between

the defected and sound areas of the component ([5], [6], [7]). More traditional damage detection methods make use of this concept in techniques involving ultrasonic thermography, ultrasonic shearography ([5]) or methods involving the application of certain time-domain transforms such as the Continuous Wavelet Transform (CWT) and the Short-Time Fourier Transform (STFT), and frequency-domain analysis (modal analysis) for manual threshold defect identification ([8]). But in recent years, the application of advanced computational algorithms, such as artificial neural networks, support vector machines and other Machine Learning (ML) and Deep Learning (DL) techniques have achieved high success for various types of fault recognition.

### **Machine Learning context with fault detection**

Machine Learning (ML) for fault detection focuses on two main areas, fault classification and anomaly detection. Fault classification presents a step further in identifying both an anomaly and which type occurred, as to anomaly detection algorithms which focus mainly on detection whether or not there exists an anomaly on the specimen [9]. Nonetheless, the use of ML techniques involves a previous feature engineering. The performance of these damage detection algorithms predominantly depends on the choice of the features and the classifier. Moreover, fixed or hand-crafted features need previous knowledge and a degree of expertise to be chosen, and may require a large computational power, which may hinder their use in real-time analysis [10]. Not meaning implicitly better performances or results, Deep Learning (DL) techniques offer a degree of feature learning, from which an algorithm learns a transformation or sequence of transformations of the raw data.

In previous years, impressive advances have been made in the field of ML and transitioning emerging DL applications for Non-Destructive Testing (NDT) have been more recently surging [11]. The development of such DL techniques appears from image recognition and classification contexts, being that a considerable amount of architecture designs for artificial neural networks have been created and achieved succeeding more accurate results ([12], [13], [14]). Nonetheless, the use of deep neural networks has spread to the mechanical and aeronautical industries for several applications such as signal analysis, image classification and more.

Regarding signal analysis, deep neural networks offer a solution for feature extraction and recognition which has been used for damage detection in beam structures ([15], [10]), for fault recognition in rotating machinery and motor fault detection ([16], [17], [9], [18]), for psychoacoustic analysis ([19]) and other non-engineering related studies such as the electrocardiogram classification ([20]). Moreover, feature extraction techniques have been applied on time-signals to generate images which can be evaluated with DL algorithms for the same Structural Health Monitoring (SHM) applications ([21], [11]). Linking damage detection with image recognition, studies have been done outside of signal analysis fields, to use these deep neural networks for wind turbine fault classification ([22]) and autonomous crack detection ([23]).

In this thesis project, within the context of Non-Destructive Testing (NDT), Machine Learning (ML) and Deep Learning (DL) techniques were applied with the purpose of damage detection in composite panels of different nature. Containing both a simulation and experimental component, this work linked such techniques into functional algorithms developed to evaluate such data as mode shapes, direct

time-signals and Frequency Response Functions (FRFs), and other feature extraction results from the Continuous Wavelet Transform (CWT) and Short-Time Fourier Transform (STFT). In the simulation component, several plates were modulated and evaluated by such algorithms, in order to build a robustness analysis, from which, considerations regarding limit damage thickness detectability, were concluded. For the experimental part, both time and frequency data retrieved from the Laser Doppler Vibrometer (LDV) were used to build damage detection algorithm methodologies, which will be compared between themselves, to take conclusions regarding the best type of algorithm to apply with this project's purpose. Additionally, an optimization step was performed to the DL algorithms, with the aim of obtaining optimized parameters and an optimized implementation of such techniques.

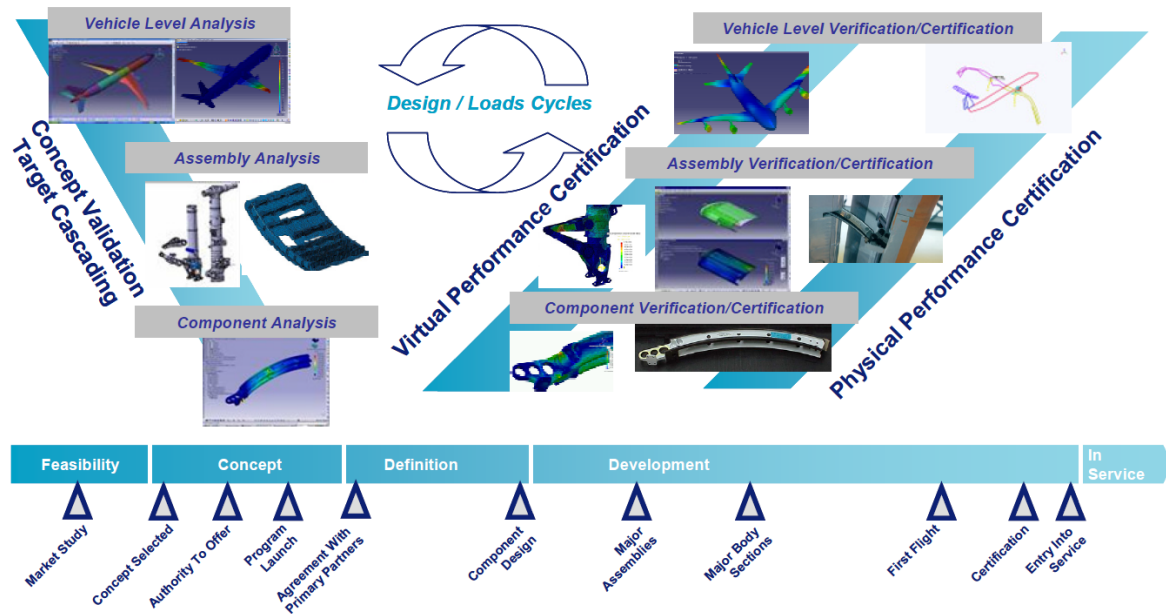
## 1.2 Motivation

The aeronautical industry grows yearly with the increase of global population able to afford travelling by air, which relates to the unparalleled increase of passengers for the airlines. To cope with this annual expansion of passengers, the main aircraft manufacturers ramp up their productions, which comes accompanied with the scale up of maintenance and control operations. In 2019, the world fleet is estimated to be consistent of close to 28 thousand aircraft and this number is expected to rise up to 40 thousand by 2029 [24].

The net-profit of this industry is estimated to have been around \$36 billion in 2019, coming up from \$32 billion in 2018. However, these values are both lower than the profit in 2017 (\$ 38 billion). This is mainly due to the increase in jet fuel prices and the fact that airlines are keeping older aircraft in the fleet, which are less fuel efficient, and thus also has an impact in greenhouse gas emissions. Moreover, as there is an expansion of business in commercial aviation, there is also an expansion of the maintenance, repair and overhaul (MRO) market which is estimated to increase by 3.5 percent over the next decade. In 2018 these MRO costs ranged in the \$70 billion, which increased to \$82 billion in 2019, representing around 9 percent of total airline operational costs [25].

These costs are divided in the many steps of validation and verification through the different levels of component, assembly and full vehicle involved in the manufacturing of an aircraft (figure 1.1). The increased use of lightweight materials like composites, aluminium alloys and titanium are expected to lower these MRO costs since they reduce the need for corrosion or fatigue inspections. This reduction can achieve an overall 60 percent margin for aircraft using these more advanced materials, which can also bring advantages in reducing its operational carbon footprint and greenhouse gas emissions [24]. However, the more complex aircraft design also implies additional testing requirements. For example, for Ground Vibration Testing, a certification procedure typically performed very late in the development process, this bring challenges of increased testing complexity and need for high results accuracy, while maintaining a cost-effective certification procedure [26].

All in all, lightweight materials possess unique properties and are highly used amongst the aerospace and the automotive industries. Thus being obvious subjects of thorough studies into classifying their behaviour and, more related to this project, into damage detection. Guaranteeing a methodology capable



**Figure 1.1:** Aircraft development process [26].

of identifying damages in the early stages of manufacturing and use, within a production line, represents the prevention of major economical dispenses and malfunctions. Therefore, this thesis project aimed for the development of a fast classification methodology of defects in composite plates, along with the detection of their location and shape.

In order to achieve such a methodology without the need of manual threshold selections, Machine Learning (ML) and Deep Learning (DL) techniques were implemented in algorithms made to analyse different types of data and outputting a defect classification. With Machine Learning (ML), high-performance algorithms were developed, but which required previous feature extraction from time-domain and frequency-domain methods (CWT, STFT and modal analysis). With Deep Learning (DL), end-to-end models were created with automatic feature extraction and classification, however at a high computational cost to train. Such algorithms like Convolutional Neural Networks (CNNs) require a considerable amount of time to train, but once trained, provide for a fast classification tool. All in all, implementing Artificial Intelligence in the damage detection procedures for composite plates also provided for the achievement of high accuracies.

The most difficult barrier this project had to deal with, regarding the implementation of Artificial Intelligence, was the need for many hyperparameter tuning, especially for the CNNs' training. With this motivation, a Bayesian Optimization methodology was developed aiming for automatic tuning of such hyperparameters, based on best performances. Thus obtaining optimized trained models that provide high accuracies in the damage classification task.

In the end, within this project, successful damage classification algorithms were created for both time and frequency-domain types of data. These algorithms overcome the necessity of a manual parameter or threshold manipulation, thus having been able to be applied on multiple plate studies.



## 1.3 Objectives

- Understand the fundamentals of damage detection procedures based on the Local Defect Resonance (LDR) concept, through different metrics, contacting with the state-of-art commercial software;
- Conduct multiple measurements on different plates, in order to more robustly test developed algorithms;
- Review the knowledge on unsupervised learning algorithms, namely k-means clustering and gaussian anomaly detection;
- Review the fundamentals for feature extraction from time signals and Frequency Response Functions (FRFs) using Convolutional Neural Networks (CNNs);
- Create unsupervised learning algorithms both for time and frequency-domain types of data, capable of detecting the presence of damages in the test plates;
- Using Convolutional Neural Networks (CNNs), train models able to extract features from time and frequency-domain data for feature classification dedicated to damage detection;
- Perform a comparison of the developed algorithms based on detectability results for the same dataset measured on a single plate.

## 1.4 Thesis Outline

This thesis project text is divided into three parts: a theoretical part presenting the concepts this project was based on; an experimental part containing the developed algorithm methodology for the experimental data obtained from the Laser Doppler Vibrometer (LDV); a simulation part containing the developed algorithm methodology for the simulation data obtained with the Siemens Simcenter 3D<sup>®</sup> software. Part II (Defect Detection for Experimental Data) is divided into three chapters. A first one to describe the experimental setup, and two following dedicated to the time data and frequency data procedure, since from the Laser Doppler Vibrometer (LDV) both time and frequency data can be retrieved. Part III (Defect Detection for Simulation Data) is divided into two chapters. Firstly, a chapter to explain the simulation setup, and a following dedicated to the SOL 103 - Real Eigenvalues procedure of data acquisition and algorithm methodology. Finally, one chapter containing other results obtained with this project's algorithms and one chapter with concluding remarks are presented.

## **Part I**

# **Theoretical Background**

## 2

# Local Defect Resonance

Local Defect Resonance (LDR) makes use of high frequency vibrations to get a localized resonant activation of the defect. This concept is based on the fact that the inclusion of a defect leads to a decrease in stiffness for a certain mass of material in this area ([5], [6]). This phenomenon manifests itself in a particular natural frequency which can be defined as the LDR fundamental frequency.

Therefore, this technique represents an extension of traditional modal analysis frequency regimes to higher ones. The high frequencies associated with the localized defect activation provide high contrast between the defected and sound areas of the component [7]. The defect's response measured at its resonance frequency highly surpasses the one obtained at the natural frequencies of the component. This strong wave interaction causes a strong rise of local temperature and non-linear behaviour in the frequency band of the defect's response [27]. These aspects have been used to improve defect localization and evaluation through ultrasonic thermography, ultrasonic shearography methods. Moreover, the induced LDR deformation of the defect's features dissipates heat through viscoelastic damping, thermoelastic damping and plastic deformation. Regarding the latter, some considerations must be made to avoid this destructive nature, in order not to damage the components for non-destructive testing purposes [28].

In many studies, a Laser Doppler Vibrometer (LDV) is used to measure the out-of-plane vibrational surface response, thus is able to capture the high vibrational patterns of a LDR. Nonetheless, this defect vibration phenomena also exist in in-plane directions. This thesis project used the referred measuring device to perform experiments on defected plates containing Flat Bottom Holes (FBHs). Even though not using more traditional methods of vibrothermography, artificial intelligence techniques were employed as processing methods to detect such defects from measurements in the non-defected surfaces of the plates, where visual inspection tells nothing. By using the LDR concept, considerations regarding its location and shape were able to be obtained.

# 3

## Machine Learning

Machine Learning (ML) was defined by Tom Mitchell [29] as:

*"A computer is said to **learn** from experience  $E$  with respect to some task  $T$  and some performance measure  $P$ , if its performance on  $T$ , as measured by  $P$ , improves with experience  $E$ ."*

To further explain the definition of ML more clearly, an example will be used. This will be the example of programming a checkers playing program, which is trained by playing thousands of games against itself and learning from each game what are the best moves to make.

The *experience  $E$*  will be the experience of playing thousands of games against itself. Therefore, quantifies the source and quantity of data used by the algorithm to learn some task.

The *task  $T$*  will be the task of playing checkers. It describes the purpose to which the algorithm is made and its ability to accomplish that purpose will be as good as its result on the performance measure.

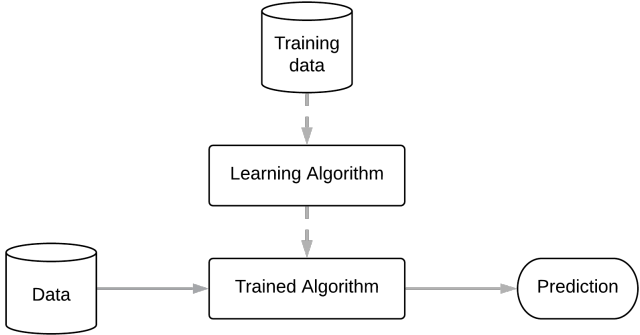
The *performance measure  $P$*  will be the probability that it wins the next game of checkers against a new opponent. This reflects the quality to which the program learned its task with its experience source.

In this chapter, some basic concepts of machine learning are going to be described, as well as the ML algorithms used for this thesis project. These are K-means clustering and Anomaly Detection using Multivariate Gaussian Distribution.

### 3.1 Basic concepts

The practical use of a ML algorithm includes two phases. In the first phase, the algorithm is trained and, in the second phase, the algorithm is used to make predictions. The training involves a set of data from which the algorithm is going to learn a certain task and the prediction involves a set of data to be

analysed by the algorithm which will then output a prediction that ranges from the many purposes to which a ML algorithm can be used to (figure 3.1).



**Figure 3.1:** Machine learning algorithm process.

The many purposes or learning problems can also be distinguished based on the characteristics of data given for the algorithm to learn. The main learning algorithm classes are Supervised Learning, Unsupervised Learning and Reinforcement Learning. In this thesis, algorithms of Supervised and Unsupervised Learning were used. The overall difference between them is how the algorithm learns. In Supervised Learning, the algorithm is taught how to do something and in Unsupervised Learning, the algorithm will learn by itself. A further description of these classes of learning problems is presented below.

**Supervised Learning:** The algorithm is going to be fed with labelled data, meaning that the data from which the algorithm learns contains the input features and labelled outputs from which its predictions have purpose for. The aim is for the algorithm to develop a function that correctly classifies the training set's labelled outputs based on the input features, being also robust in order to classify unseen data correctly. Moreover, supervised learning can also be split in classification and regression problems, based on the restrictions to the algorithm's prediction.

In *Classification* the training data is labelled for certain number of finite classes. Therefore, the predictions of the algorithm are restrained to these classes. For example, the classification of "spam" or "non-spam" emails.

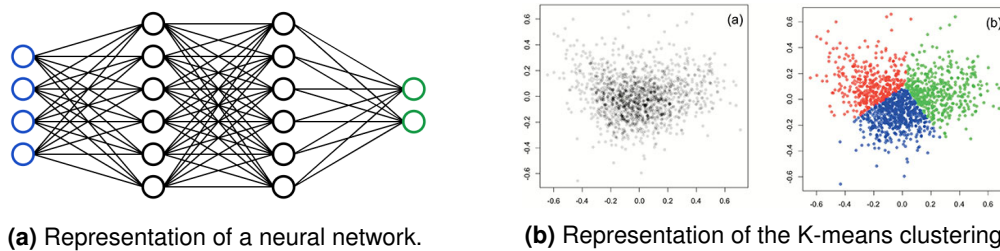
In *Regression* the prediction values are on a continuous interval, therefore can be a value different from the ones contained in the training set's labels. For example, the prediction of a house's price, based on such input features like its size, number of rooms, etc.

Some examples of Supervised Learning algorithms are: linear regression; logistic regression; neural networks; support vector machines.

**Unsupervised Learning:** The algorithm is going to be fed with unlabelled data, therefore containing only input features from which the algorithm should find structure in the data. This process is done based on clustering methods, from which training examples with similar properties get grouped into the same cluster. For example, training an algorithm for market segmentation, or

astronomical data analysis.

Some examples of unsupervised learning algorithms are: k-means; principal component analysis; hierarchical clustering; self-organizing maps.



**Figure 3.2:** Representation of a a) supervised learning and b) unsupervised learning algorithms.

## 3.2 K-means Clustering

The K-means clustering is an unsupervised learning centroid based clustering algorithm. As unsupervised learning, its method works on an unlabelled training set, where each training example contains a certain amount of features, but no labels. As a centroid based clustering algorithm it divides the training set into  $K$  different clusters, which is thus an input to this algorithm. This is one of the most famous clustering techniques since it is relatively simple to implement. Its applications ranges through many fields, like market segmentation, social network analysis, astronomical data analysis and computing clustering organization.

### 3.2.1 Algorithm

Considering a dataset with  $m$  unlabelled training examples  $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$ , where each example is a  $n$  sized vector of features ( $x^{(i)} \in \mathbb{R}^n$ ), the aim of this algorithm is to aggregate the training examples into a pre-defined amount of clusters ( $\mu_1, \mu_2, \dots, \mu_K \in \mathbb{R}^n$ ). This process can be described in the following steps:

1. Random initialize  $K$  cluster centroids;
2. Repeat until all centroid positions remain constant {

(a) Cluster assignment

for  $i = 1 : m$

$$c^{(i)} := \text{index}(\text{from } 1 \text{ to } K) \text{ of cluster centroid closest to } x^{(i)}$$

(b) Move centroid

for  $k = 1 : K$

$$\mu_{(k)} := \text{average of points assigned to cluster } K$$

}.

## Random initialization and cost function

As stated in the methodology above,  $K$  clusters must be randomly assigned (step 1) before the iterative process. This is done by randomly choosing  $K$  training examples and assigning the clusters to those points ( $K > m$ ), which implies some variability to this algorithm. It can happen that the clusters are luckily assigned near a global optima solution, or it can happen that the clusters are assigned to close points or near a local optima solution, which will freeze the iterative process of the algorithm. Therefore, this algorithm should be run for many random initializations and the best result should be chosen and identified as being the one which minimizes the cost function value  $J$ .

$$J(c^{(1)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K) = \frac{1}{m} \sum_{i=1}^m \|x^{(i)} - \mu_{c^{(i)}}\|^2 \quad (3.1)$$

Where;

$\mu_{c^{(i)}}$  : cluster centroid of the cluster to which the example  $x^{(i)}$  has been assigned.

This suggested methodology can be represented as;

1. Test algorithm for a chosen number of random initializations;

for  $it = 1 : 100$

Random initialize K-means

Run K-means. Get  $c^{(i)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K$

Compute cost function  $J(c^{(i)}, \dots, c^{(m)}, \mu_1, \dots, \mu_K)$

2. Pick clustering that provided the lowest cost.

### 3.2.2 Visualization

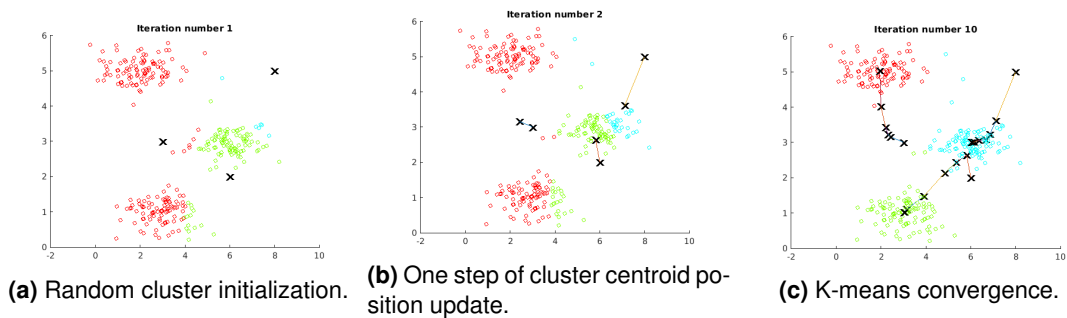
As stated before, K-means is an iterative algorithm, constantly updating the position of the clusters' centroids until convergence. In this chapter, a visual demonstration of this ML algorithm will be presented, in order to understand better its procedure. This visualization was obtained by a practical implementation of the K-means algorithm on a 2D dataset (each training example contains two features), with 3 clusters.

Once  $K$  (number of clusters) is chosen, which in this case corresponds to 3, the first step is to randomly initialize these clusters into  $K$  different training set's points. Once initialized, by computing the nearest cluster to each training example and attributing that closest cluster to that example, each cluster will now contain a group of examples which can be seen in figure 3.3a as matching colors. By analysing the position of the cluster centroids in comparison to the dataset, can be concluded that the automatic clustering of training examples seems very unbalanced.

The next step is to compute the average position of the training examples assigned to each cluster and assign it as the new cluster centroid position. This process will involve the centroids movement and therefore an updating of the new cluster centroid closest to each training example. Figure 3.3b

represents the centroid movement with lines and also shows the new clusters with the corresponding colors.

This step of cluster centroid position update is to be repeated until convergence, this is, until the cluster centroids' position remains constant with a new iteration. This will mean that an optima, either local or global has been achieved. Figure 3.3c represents the convergence of this example after 10 iteration.



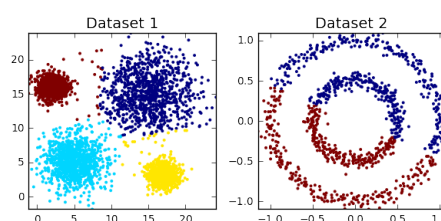
**Figure 3.3:** K-means Clustering progression.

Looking into this example's clustering final result in comparison to the dataset, there can be concluded some success as to the identification of three clusters. This dataset was by itself prone to be successfully clustered, but this algorithm doesn't do well for all kinds of datasets, for example detecting non-spherical clusters.

### Limitations

In this subsection the limitations of this algorithm which have been described or stated before, will be summarized and also visualized:

- The final convergence has a strong dependence on the initial random cluster centroid initialization, especially when the training set doesn't contain many examples;
- Choosing the number of clusters, in case it is not obvious from the application's purpose, can be challenging;
- Not all datasets will provide good clustering results with this algorithm, especially datasets that don't imply a spherical clustering as the one seen below.



**Figure 3.4:** K-means limitations to clustering non-spherical datasets.



### 3.3 Anomaly Detection based on Multivariate Gaussian Distribution

Anomaly detection is a commonly used type of ML for when the datasets contain a very unbalanced number of examples between each class. It is used mainly for unsupervised learning problems, but contains some aspects of supervised learning. Therefore, it isn't completely independent from previous knowledge on the problem such as the K-means algorithm.

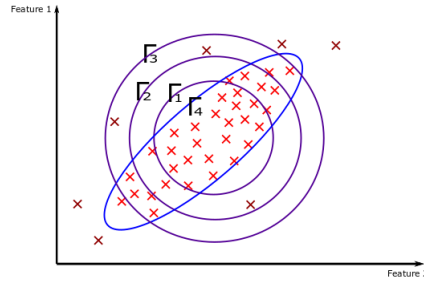
The main reason behind applying such techniques as to applying supervised learning lies in not having a relatively balanced number of examples per class. And so not being able to train supervised learning algorithms like neural networks on an unbalanced set, since they would become biased to classify the class with the largest amount of training examples. For example, training an algorithm to detect credit card frauds as to email spam classification. The latest is a problem more suitable for supervised learning since there would be enough positive examples for an algorithm to be accurately trained to classify them.

The algorithm of anomaly detection used in this thesis project relies on fitting a Multivariate Gaussian distribution to the dataset. It is important to understand these statistical concepts in order to understand the algorithm. Therefore, some concepts regarding this statistical approach will be presented along with the algorithm's description.

#### 3.3.1 Algorithm

Summarizing this algorithm's methodology, a Multivariate Gaussian Distribution is fitted to the training set's features and through a threshold selection process, the best threshold is obtained to separate training examples that correspond to anomalies from training examples that represent common and acceptable values. Since it is important to have knowledge on the Multivariate Gaussian continuous probability distribution, below some considerations regarding it will be given.

Applying a Multivariate Gaussian distribution has advantages comparing to applying a simpler Gaussian distribution. The latter has a circular assumption of probability and thus the threshold curves from which examples are separated into anomalies or non-anomalies can only be perfect circles. Hence, for a two-dimensional (2D) dataset, which means that each training example contains two features, the threshold curves that can be identified with this distribution, in reference to picture 3.5, are  $\Gamma_{1-3}$ . It could never develop an elliptical threshold curve such as the one represented by  $\Gamma_4$ . Looking at the dataset represented in picture 3.5, a circular threshold curve and so a Gaussian distribution application means that either a considerable number of non-anomaly examples (light red) will wrongly be classified as anomalies (dark red), or some anomalies will be classified as non-anomalies. Using a Multivariate Gaussian distribution, an elliptical threshold curve like  $\Gamma_4$  can be achieved, which will create a more perfect decision boundary. Hence a better anomaly detection algorithm.



**Figure 3.5:** Multivariate Gaussian vs Gaussian distribution threshold curves.

### Multivariate Gaussian distribution

Equation 3.2, characterizes the density estimation  $p(x)$  for a Multivariate Gaussian distribution.

$$p(x, \mu, \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right) \quad (3.2)$$

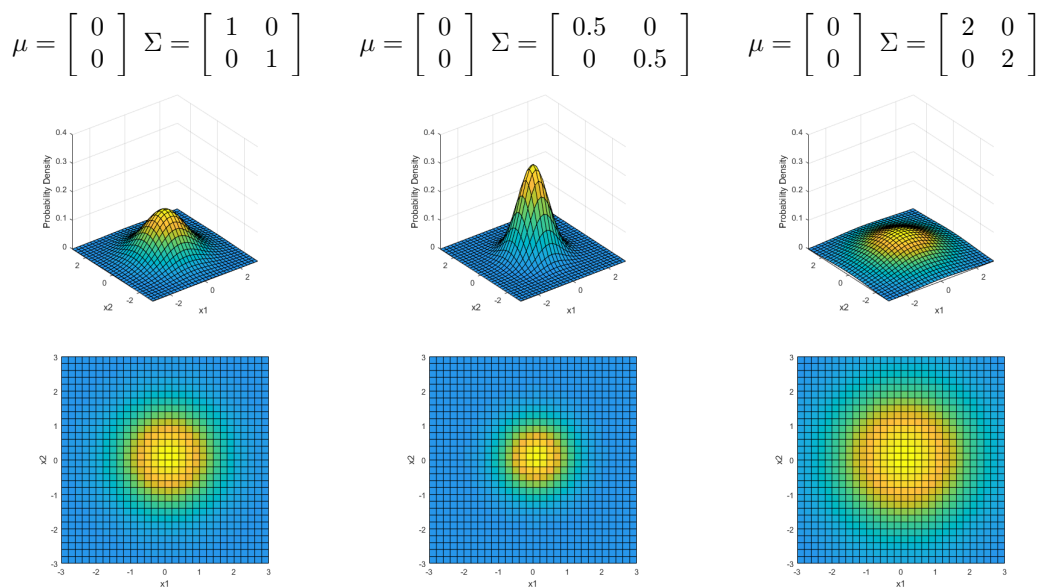
Where;

$$\mu = \frac{1}{m} \sum_{i=1}^m x^{(i)} \quad : \text{mean vector of a feature's distribution } (\mu \in \mathbb{R}^n).$$

$$\Sigma = \frac{1}{m} \sum_{i=1}^m (x^{(i)} - \mu)(x^{(i)} - \mu)^T : \text{covariance matrix } (\Sigma \in \mathbb{R}^{n \times n}).$$

With the variance of  $\Sigma$  so will vary the shape, width and orientation of the threshold boundary or contours. Nonetheless  $\Sigma$  will always be a symmetric matrix. The images below represent how the parameters of  $\mu$  and  $\Sigma$  both influence the probability density of this distribution. In order to be able to plot this variation in a 3D plot, an example of only two features was considered. This study will be presented in the format of tables containing both the Multivariate Gaussian distribution parameters and the plot of that respective distribution.

**Table 3.1:** Influence of covariance matrix's equal diagonal values

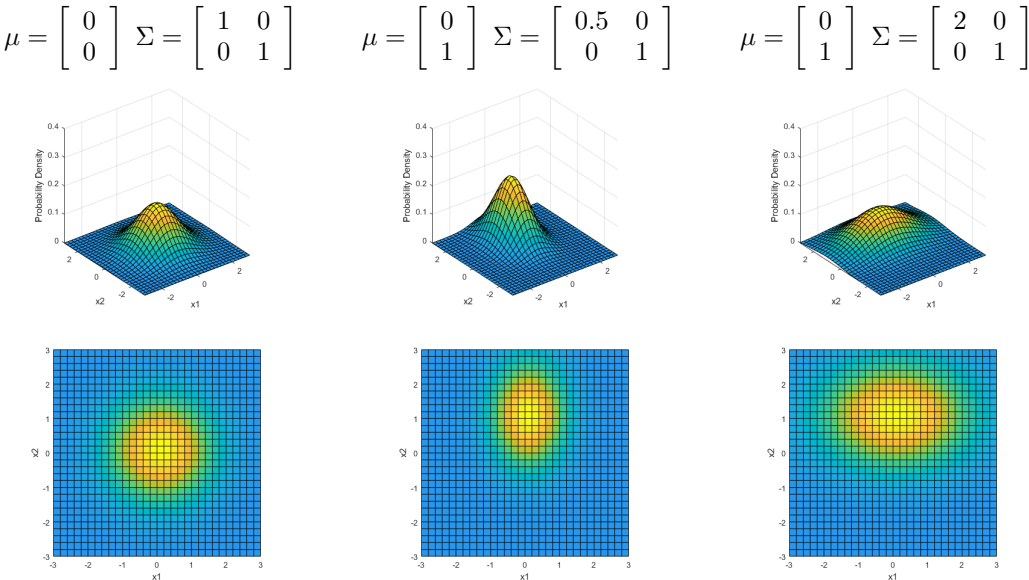


The table 3.1 portrays three different distribution and the influence of equally changing both diagonal

parameters, from the covariance matrix ( $\Sigma$ ). The reference, which is portrayed as the example to the left, contains a ( $\Sigma$ ) equal to the identity matrix. Thus it will be used as a comparison basis to the other examples. Moreover, for all of the examples above, the mean vector ( $\mu$ ) is maintained to zeros and as a result, the maximum of each distribution (yellow-est point) is located directly above this coordinate in the 2D plane represented in the third row of the table.

By comparing the second and third column results with the reference column (1st), the diagonal values were respectively decreased and increased. As a direct result the diameter of the distribution was respectively diminished and increased. The covariance matrix ( $\Sigma$ ) measures the variance or variability of the features. Thus, if its values are diminished, so will the distribution diminish in diameter. Moreover, since the integral of the volume under the surface is equal to 1, the distribution will increase in height/value, as can be seen in the middle picture of the second row. The contrary is also applicable, increasing the diagonal values of  $\Sigma$ , so will increase the variance and therefore, the distribution will become flatter and wider.

**Table 3.2:** Influence of covariance matrix's different diagonal values and the mean vector's values



The table 3.2 shows once again the reference case where  $\Sigma$  is equal to the identity matrix and  $\mu$  is centred to the origin. It also shows two more cases that illustrate the influence of the mean vector ( $\mu$ ) and unequal values in  $\Sigma$ 's diagonal. Looking at  $\mu$  for the second and third column cases, its second value was changed to 1 in comparison to the reference. As a result, the center of the distribution (yellow-est point) was moved to a point in the 2D plane shown as the third column, which corresponds to (0, 1).

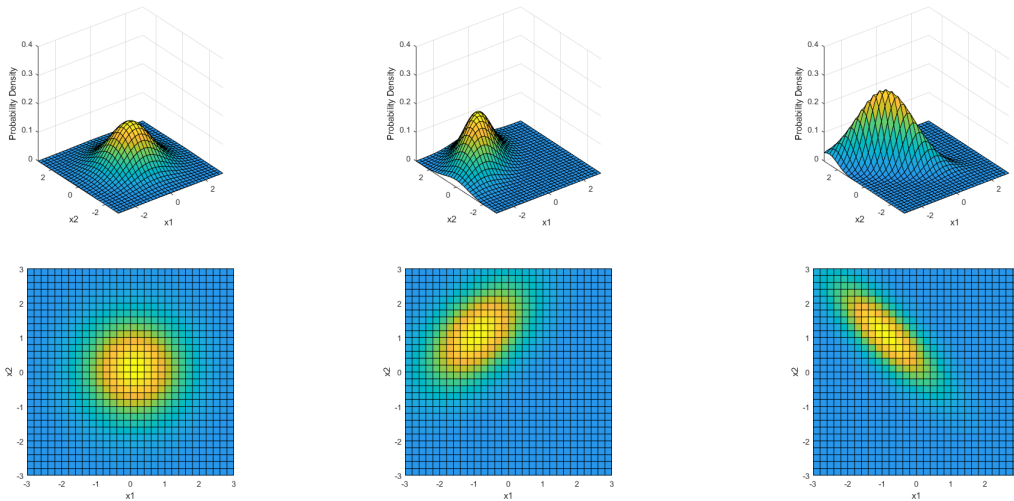
For both cases in the second and third columns, the first value of the covariance matrix was changed, for the second column decreased and for the third column increased. In terms of physical meaning, for the second column, the variance was decreased for the first feature and for the third column, the variance of the first feature was increased. Therefore, with respect to the axis representative of the first feature (xx axis), the width of the distribution was respectively decreased and increased. This phenomenon can be clearly seen by the last row's pictures. Moreover, as the width of the distribution decreases, so will the

height increase, since, as stated previously, the integral of the volume under the surface must maintain equal to 1.

In comparison to the pictures in table 3.1, it can be observed that these distributions are no longer circular. Therefore, the property of Multivariate Gaussian distribution in comparison to a simple Gaussian distribution of being able to create non circular boundaries has been demonstrated.

**Table 3.3:** Influence of covariance matrix's non diagonal values and the mean vector's values

$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \mu = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix} \quad \mu = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix}$$



As the previous cases, table 3.3 contains a first column with the reference case, where  $\mu$  equals to a zero vector and  $\Sigma$  equals to the identity matrix. In comparison to table 3.2 the mean vector ( $\mu$ ) has now its two values different from one. As such, the center (yellowest point) of the second and third column distributions are displaced both in the x-axis and y-axis. This can be clearly seen from the pictures in the third row of the table.

Considering the differences in the  $\Sigma$  matrix, for this table, the non-diagonal values were also studied. In the second column case, these values were set to 0,5 and in the third column case, they were set to -0,8. The covariance matrix must be symmetric and therefore, both non-diagonal values must be equal. The difference from having these values positive like the second case, or negative like the third case lies on the orientation of the distribution. By observing the third row images on the table, the orientation of the distribution can clearly be seen.

By increasing the non-diagonal entries from 0,5 to 0,8, the distribution gets more thinly peaked, less wide. With a high value like 0,8, all probability within the Gaussian distribution lies on the narrow region. Moreover, the contribution of existing non-diagonal values different from zero lies on the distribution shape being diagonal, unlike the cases on the previous two tables.

The fact that this probability distributions can vary in shape, considering the covariance matrix calculation, provides an important property to this algorithm: the ability to correlate the features being evaluated for each training example. This means this algorithm has a feature analysis capability, which is important because it provides some liberty as to the choice of features and not needing to correlate

them manually.

### Application to Anomaly Detection

Considering the information above mathematically demonstrated for fitting a Multivariate Gaussian distribution to the training set's features, the steps of the Anomaly Detection algorithm can be described as:

1. Choose features  $x_j$ , that might be indicative of anomalous examples;
2. Fit a Multivariate Gaussian distribution by calculating the mean vector  $\mu$  and the covariance matrix  $\Sigma$ ;
3. Given a new example  $x$ , compute  $p(x)$ ;
4. Identify an anomaly if  $p(x) < \epsilon$ .

Where  $\epsilon$  represents a constant threshold value from which an example's probability value may be assigned to an anomaly, if it is below this threshold.

$$p(x) < \epsilon \Rightarrow x \text{ is an anomaly}$$

$$p(x) \geq \epsilon \Rightarrow x \text{ is NOT an anomaly}$$

### Threshold ( $\epsilon$ ) automatic selection

In the end, after fitting the distribution, the threshold will be the selection parameter, therefore the success of this algorithm's classification depends highly on this value. If it is too high, the algorithm will have a tendency to classify more examples as anomalies and therefore may misclassify non-anomalies as anomalies. If it is too low, the algorithm will have a tendency to classify anomalies as non-anomalies.

A manual selection can be applied which would imply visualizing the results in a step-by-step procedure, what can be labour-some and not practical. A better solution exists through creating an automatic threshold procedure methodology, by analysing a cross-validation subset of the overall dataset. Using this subset of data, through applying metrics of evaluation for various threshold results, a best threshold can be selected as being the one providing the best anomaly classifications. For example, considering a dataset with 5000 examples and only 10 anomalies within, a following dataset division could be used.

$$\begin{array}{l}
 \text{dataset} \Rightarrow \begin{cases} 4990 & y = 0 \\ 10 & y = 1 \end{cases} \\
 \begin{array}{l} \rightarrow \text{train} \Rightarrow \{ 3000 \quad y = 0 \\ \rightarrow \text{C. V.} \Rightarrow \begin{cases} 995 & y = 0 \\ 5 & y = 1 \end{cases} \\ \rightarrow \text{test} \Rightarrow \begin{cases} 995 & y = 0 \\ 5 & y = 1 \end{cases} \end{array}
 \end{array}$$

Since datasets similar to the one presented above are have skewed classes, a common evaluation metric like accuracy wouldn't provide good insights. For example, if a certain threshold on an anomaly detection algorithm applied on the dataset above, would classify all examples as non-anomaly, the accuracy would still be 99.8 %. Hence, other metrics should be used and in order to present them, the concept of confusion matrix, or error matrix will be introduced.

**Table 3.4:** Confusion matrix

		Actual	
		y = 1	y = 0
Predicted	y = 1	True Positive	False Positive
	y = 0	False Negative	True Negative

The confusion matrix considers 4 classifications based on the correspondence between the predicted values and the ground truth. Considering the problem of anomaly detection being described and for future reference, the value 1 applies to anomalies and 0 non-anomalies. Therefore a *False Negative* interpreted in this algorithm's context means that the algorithm classified an example as non-anomaly where in reality it corresponded to an anomaly. The *Trues* mean prediction success and the *Falses* prediction error. Based on this proposed classification, the following metrics can be calculated:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad (3.3)$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \quad (3.4)$$

And for reference to the common accuracy principle:

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Example\ Number} \quad (3.5)$$

*Precision* is a measure that quantifies, in this algorithm's context, out of all predicted anomalies, what fraction actually was an anomaly, whereas *Recall* quantifies, out of all anomalies, what fraction was correctly predicted. In terms of classifying the success of an algorithm, it is at the same time important to be precise in the predictions made, without missing out on many anomaly examples. Therefore, there is another metric that combines both *Precision* and *Recall*, which is identified as the best metric to use in these anomaly detection algorithms, called *F<sub>1</sub> score*.

$$F_1\ score = 2 \frac{Precision * Recall}{Precision + Recall} \quad (3.6)$$

In order to make the threshold selection problem automatic, a methodology can be designed using the *F<sub>1</sub> score* to evaluate several threshold ( $\epsilon$ ) values.

for  $\epsilon = [low\ limit; high\ limit]$

1. anomalies =  $p(x) < \epsilon$
2. calculate *F<sub>1</sub> score*

3. if this threshold's  $F_1$  score is higher than a the previous, save this threshold as new best

In the end, the best threshold values will be calculated form the cross-validation subset of examples. This part of the Anomaly Detection algorithm provides the similarities to supervised learning, since a previous knowledge on the cross-validation's example labels is necessary to calculate the  $F_1$  score.

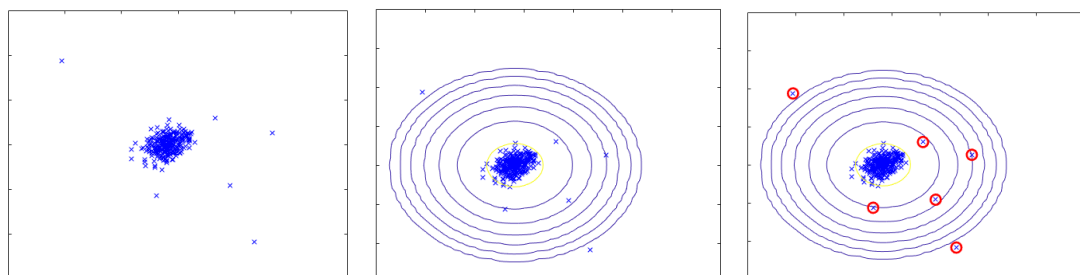
### 3.3.2 Visualization

The Anomaly Detection algorithm receives a dataset, applies a Multivariate Gaussian distribution to all examples, calculates the best threshold with an iterative process and applies it to the probability distribution in order to separate the anomalies from the non-anomalies. In this section, a visualization of the working procedure of this algorithm will be presented.

In order to visualize the working procedures of this algorithm, a two-dimensional (2D) dataset will be considered. Its plot is shown in the figure 3.6a. By analysing it, it's possible to see a group of examples in the center of the picture and some example outliers separated from this main group. These correspond to examples whose features vary largely from the usual feature values. For demonstration purposes, the following images were cropped of the axis values and axis variables.

The first steps of algorithm application, as written in the subsection 3.3.1 is to fit a Multivariate Gaussian distribution to the training set, by calculating  $\mu$  and  $\Sigma$ . The figure 3.6b shows the contours of the fitted Multivariate Distribution. It's possible to see that most examples are contained to the high probability elliptical region, while the anomalous examples are in the regions with lower probability.

The next step of the algorithm is to calculate the best threshold on an iterative process using a cross-validation subset, on which there is previous knowledge on the example's labels as anomalies or non-anomalies. Once that iterative procedure has finished, this threshold value will be used to separate the original dataset which is being plotted in this section, into anomalies and non-anomalies. As can be seen in figure 3.6c, the algorithm detected all anomalies out of the high probability region, which are demonstrated in the picture by red circles.



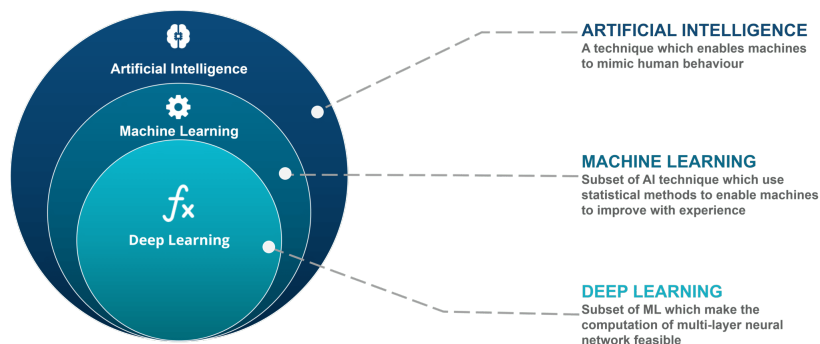
(a) 2D dataset for Multivariate Gaussian Anomaly Detection (b) Multivariate Gaussian distribution contours (c) Outcome of anomaly detection algorithm

**Figure 3.6:** Anomaly Detection progression.

# 4

## Deep Learning

Deep Learning (DL) is a subset of Machine Learning (ML) techniques highly focused on artificial neural network algorithms. Thus, the same basic concepts that apply to Machine Learning (ML), which were described in section 3.1, also apply for DL in terms of procedure and classes of learning problems. DL involves a step further in complexity and also in terms of power to represent more complex functions. Such architectures like deep neural networks, convolutional neural networks, recurrent neural networks are deep learning algorithms that have been applied to many purposes and fields like computer vision, defect detection, audio and speech recognition, bioinformatics and so on.



**Figure 4.1:** Deep Learning and Machine Learning panorama inside the Artificial Intelligence area.

### 4.1 Convolutional Neural Networks

*Convolutional Neural Networks* (CNNs) are specialized artificial neural networks, that employ the mathematical operation of *convolution*. This is the defining difference from fully connected neural networks, where general matrix multiplication is applied instead. The origin of CNNs goes back to image recognition (firstly for digit identification [30]) and many important advancements with ground-breaking architectures have been proposed on ImageNet Large Scale Visual Recognition Challenge (ILSVRC). However, CNN application is not confined to 2D image recognition, having been also implemented for 1D and 3D data analysis, for many purposes across multiple fields [31].



The architecture of a typical Convolutional Neural Network (CNN) is structured as a series of stages. The first stages contain usually two main types of layer operations: convolutional and pooling layers. In a convolutional layer, a kernel with pre-defined dimension is convolutionally multiplied across the layer's input generating a feature map. The amount of kernels in that layer will determine the amount of feature maps obtained. With a well trained CNN to a certain purpose, those feature maps will contain characteristics of the inputs which describe its features and better help the classification or regression problem to which the CNN is being applied.



**Figure 4.2:** Sigmoid and ReLU activation functions.

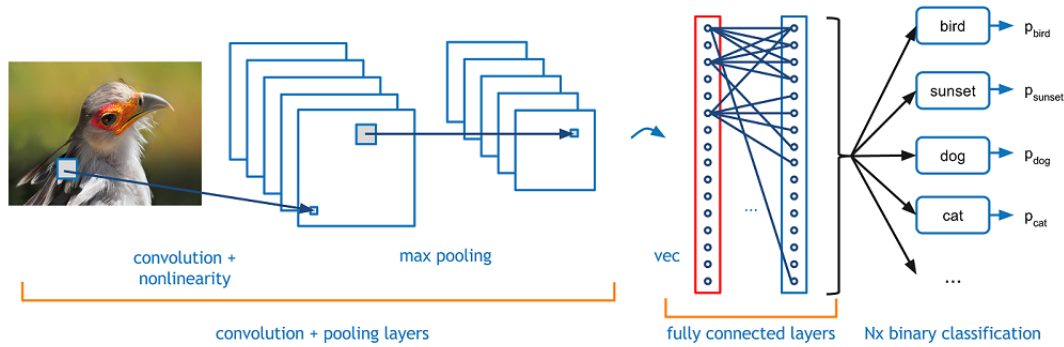
After this convolution operation, the outputs are commonly multiplied by a non-linearity, like the sigmoid-function or the Rectified Linear Unit (ReLU), before being submitted to the pooling operation. These two activation functions are represented in figure 4.2, but the most used and also used in this thesis project was the ReLU. The pooling layer performs sub-sampling associated with a certain operation that can be calculating the maximum or the average of all values within the pooling kernel. This operation reduces the dimension of the convolution operation outputs and is usually always associated to it, just like represented in figure 4.3.

Two motives stand behind this configuration. Firstly, data such as images are represented by value arrays and local groups of values which are often correlated, forming patterns that can be detected. Secondly, local statistics of images or signals are independent of the location. Therefore, if a certain pattern (like an edge) can appear in one image in a certain location, it can also appear in another image within a different location, which can nonetheless be detected using the same kernel. As described before, the role of convolutional layers is to detect local conjunctions of features from the previous layers. This requires the use of a pooling operation in between, that performs semantic merging of different feature values into one.

Moreover, a further operation can be performed in between convolution and pooling, the batch normalization. This operation normalizes the values obtained from the convolutional layer in order to train the next layer faster. Furthermore, it makes weights deeper in a CNN more robust to changes than the ones in the first layers. The values in the middle and latter layers have a lot of variance, which causes its weights to have some increased difficulty in stabilizing. By applying batch normalization, in the end, no matter the variance of values, they will have around the same mean and variance. Thus, the learning procedure is made easier and more robust for the latter layer's weights.

Usually, in a common CNN architecture, after convolutional and pooling blocks, a set of fully connected layers is used. This architecture's purpose is to detect features with the convolutional operations and after these features being detected, performing a classification or regression procedure using the fully connected layers, just like represented in figure 4.3. The number of weights in these layers can reach very big values (fully connected layers tend to have much more parameters than convolutional

ones), thus biasing problems are common to appear. In order to surpass this issue, a relatively recent regularization technique was invented, the dropout. A certain percentage of neurons along with their weights is selected to be cancelled for each back-propagation step. Working with a smaller network will have a regularizing effect, since an output neuron can't rely on anyone feature, because that feature can go away randomly. Therefore, that output unit will be more prone to distribute the weights along its input channels. This effect will shrink the squared norm of the weights, which helps prevent over-fitting.



**Figure 4.3:** Convolutional neural network architecture representation with the usual convolutional operations before the fully connected layers.

Also observing figure 4.3, there can be seen that after the fully connected layers, a classification procedure is carried out. It is common to use a softmax layer to help multi-class classifications, which makes use of the softmax function based on exponentials, to better present the probabilities of the fully connected layers' classification procedure. Considering  $z^{(l)}$  to be the output of the classification procedure previous to the softmax application, for a certain class of output:

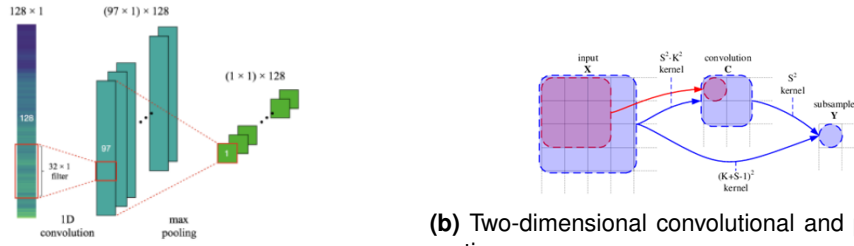
$$\text{softmax function} = \frac{e^{z^{(l)}}}{\sum e^{z^{(l)}}} \quad (4.1)$$

### 4.1.1 1D vs 2D convolutional neural networks

Depending from the type of input so depend the kernels from the CNN operations. In case one-dimensional signals are being analysed, the use of more common 2D kernels for convolutional and pooling layers cannot be applied. To flow through a vector of values representing the one-dimensional signal, a 1D kernel (vector) must be used. The most common applications of deep learning technology focus on image analysis, where the input can be represented by an array of values. For these applications, the kernels are also arrays, as represented in figure 4.4. In this thesis project both 2D and 1D convolutional neural networks were used in order to analyse images and one-dimensional vectors correspondent to Frequency Response Functions (FRFs) and time signals.

## 4.2 Transfer Learning

Transfer Learning (TL) is a Machine Learning (ML) method of reusing previously developed models for a certain task, as starting points for another task. When referring to Convolutional Neural Networks



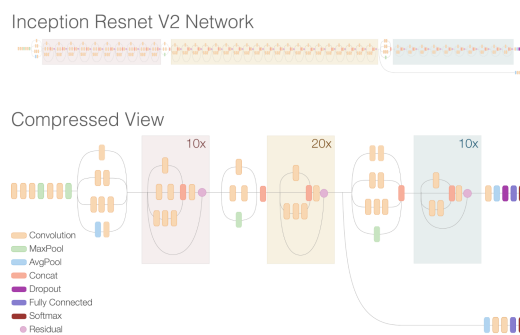
(a) One-dimensional convolutional and pooling operations.

(b) Two-dimensional convolutional and pooling operations.

**Figure 4.4:** One and two-dimensional convolutional and pooling operations comparison.

(CNNs), some complex architectures have been developed which are difficult to replicate, due to their complexity and size. Given the fact that complex and deep architectures can provide better results for certain tasks with more complex feature recognitions than shallow architectures, the re-use of these deep architectures for other task can generate better outcomes than those which would be achieved from a custom shallower CNN.

Famous architectures like Inception-ResNet-v2, DenseNet-201 can reach many modules of depth (164, 201 respectively), with complex architecture links and therefore the ability to re-use such architectures provided this thesis project with great tools for classification problems. Moreover, these nets have pre-trained layers that perform certain feature recognition operations, which may also be useful for the different task they are being applied to. The shallower the layer, the more basic features are extracted, therefore, to re-use these layers might be beneficial for a certain task, regarding optimizing the computational time and since it may not be useful to train every layer if there isn't a sufficient amount of data. Depending on the amount of data, so should depend the ratio between amount of layers frozen and trained. With the decrease of amount of data available for training, so should increase the number of frozen layers, therefore leaving a less amount of layers to be actually trained.



**Figure 4.5:** Inception-ResNet-v2 architecture.

If transfer learning is being performed for a certain classification or regression problem different from the original network's task, some alterations should be performed in the last classification layers. In this thesis project, two classes were usually used for classification problems and thus, the training of previous architectures involved the changing of the classification layer and also the last fully connected layer to 2 neurons.

# 5

## Bayesian Optimization

Deep Learning (DL) algorithms are not parameter-free, they must be tuned with hyperparameters controlling the learning rate or the capacity of the underlying models must be specified and carefully tuned. Unfortunately this tuning frequently requires more experienced knowledge, rules of thumb or even *brute-force* searches [32]. These less practical methodologies can be avoided by performing automated optimization of such parameters based on best performances. In this field, of optimization algorithms, the Bayesian Optimization has been shown by [33] to outperform other state-of-art global optimization algorithms on a number of challenging optimization benchmark functions.

Bayesian Optimization owes its name to the *Bayes* theorem, which states in a simplified way, that the posterior probability of a model  $M$ , given evidences  $E$ , is proportional to the *likelihood* of  $E$  given  $M$  multiplied by the prior probability of  $M$  [34]. The theorem is stated as:

$$P(M|E) \propto P(E|M)P(M) \quad (5.1)$$

According to [34] and [35], first  $x_i$  is defined as the  $i$ th sample and  $f(x_i)$  corresponds to the observation of the objective function,  $f(x)$ , at  $x_i$ . As observations are stored in  $D_{1:t} = \{x_{1:t}, f(x_{1:t})\}$ , the previous distribution is combined with the likelihood function  $P(D_{1:t}|f)$ . Thus, considering the optimization procedure, the objective function is assumed to be drawn from the probabilistic model:

$$P(f|D_{1:t}) \propto P(D_{1:t}|f)P(f) \quad (5.2)$$

The Bayesian Optimization finds the minimum of a certain target function,  $f(x)$ , on a bounded set  $\chi$ , which is taken as a subset of  $\mathbb{R}^D$ . The most significant difference of this method in comparison with other optimization methods is the fact that it builds a probabilistic model for  $f(x)$ , and then exploits this model to make decisions on where to move on the set  $\chi$ , in order to find the best set of parameters to test. Thus it makes use of all the information provided by the previous evaluations of the objective function, in order to find the best regions of parameters to be tested in order to obtain the minimal error and not just randomly test parameters within the bounded set  $\chi$ . The resulting procedure is capable of finding a minimal error solution for complex functions with relatively few evaluations.

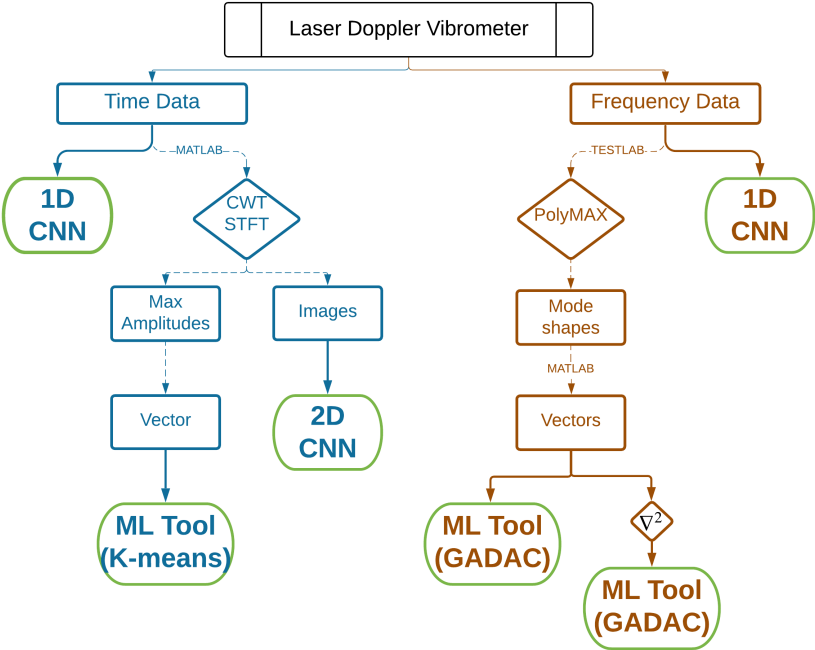
## **Part II**

# **Defect Detection for Experimental Data**

The aim of this thesis project is to develop an algorithm based methodology for damage detection in lightweight structures. Several lightweight plate specimens of Carbon-Fiber Reinforced Polymer (CFRP) and PMMA, containing Flat-Bottom Hole (FBH) damages were measured. These plates contain the defects on the bottom surface and are measured looking at the top surface, where no traces of the defects can be visualized (figure 6.2). Several algorithms were developed with ML and DL techniques in order to analyse measured data and detect the damage locations in the bottom surface of the plates.

The developed algorithm methodology is variate in terms of techniques, input data and data transformation and post-processing techniques. From the Laser Doppler Vibrometer (LDV) , both frequency and time data can be retrieved, and therefore the developed methodology is separated in time and frequency domain algorithms. The fluxogram shown in the figure 5.1 contains the sum-up of all algorithms and post-processing steps from the raw LDV measurement data. The algorithm types are associated to their names, and are always presented on the ending of each column. Therefore the algorithms are always the last step for the damage detection procedure. Five algorithms were developed, two containing Machine Learning (ML) techniques, the ML tool (K-means) and the ML tool (GADAC), and three containing Deep Learning (DL) techniques, the three Convolutional Neural Networks (CNNs).

Considering the time data procedures, all processing techniques are applied in MATLAB® . These are the Continuous Wavelet Transform (CWT) and the Short-Time Fourier Transform (STFT) whose application aims at highlighting a defect's presence. As for the frequency data processing techniques, they are performed in Siemens Simcenter TestLab® using the Polymax tool to perform modal analysis in order to obtain the mode shapes. In this part of the thesis, all damage detection algorithms will be presented, with their methodology and results individually explained. Moreover, a brief overview of all the input data will be also given, in order to have a better comprehension of the type of data being analysed by these algorithms.

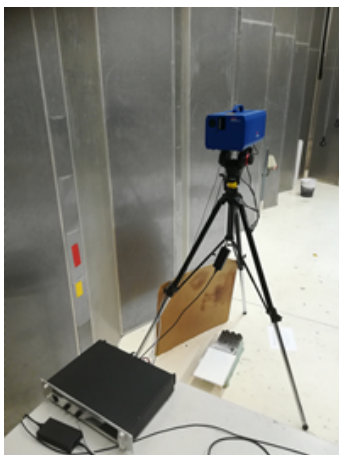


**Figure 5.1:** Algorithm methodology developed in this thesis project for experimental data.

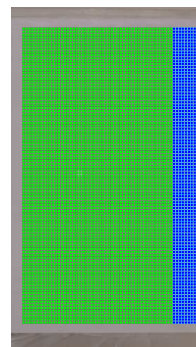
# 6

## Experimental Setup

The measurements done in this project were performed using an Optomet SWIR Laser Doppler Vibrometer (LDV) . This contactless measurement technique is based on the Doppler effect associated with the scattering of the laser beam. It is a fast and simple solution to measure the vibration velocity, eliminating the mass loading on a structure implied on mote traditional measurement instruments like accelerometers or strain gages. From these measurements, two types of data could be retrieved, Frequency Response Functions (FRFs) and time signals containing the vibration history of that specific measured point during the excitation's period. In terms of experimental setup, the LDV was pointed directly at the plate's top (undamaged) surface, which was standing on top of foam. A piezoelectric patch was glued on the bottom surface of the plate and provided the excitation. According to the Local Defect Resonance (LDR) concept, in order to excite the plate into resonance patterns dominated by defect vibration, the frequency bands used to excite the plates using the PZTs were in the order of kHz. Most of the excitation's frequency bands were up to 40 kHz, and some reached 80 kHz. Moreover, an amplifier was used to enhance 50 times the generated signal using a Falco Systems WMA-300. Figure 6.1 contains the described experimental setup.



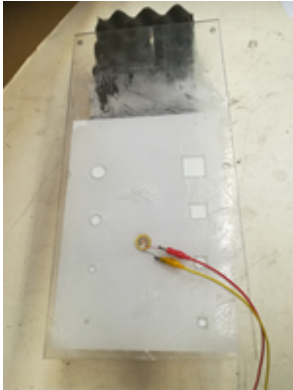
(a) Experimental setup.



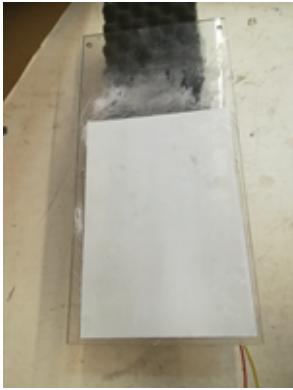
(b) Grid measured by the LDV . Blue points correspond to points not yet measured, and the green correspond to already measured points

**Figure 6.1:** Experimental setup with the LDV and the plate to be measured below over a piece of foam. To the right is presented an example of a measured grid by the LDV .

Although many plates were measured in this thesis project, for this part of the thesis, emphasis will be given to the measurements done on the PMMA plate represented in figure 6.2. This is due to comparative reasons, in order to show the performance of each of the further described algorithms in reference to a single dataset. In chapter 11, results obtained with the algorithms described in this part of the thesis are presented for other plates. The measured PMMA plate was manufactured by bonding a polystyrene sheet of  $300 \times 210 \times 0.5 \text{ mm}^3$  onto a  $5 \text{ mm}$  thick PMMA plate containing square and round holes of different sizes, using epoxy resin. It was excited with the piezoelectric patch up to a frequency of  $40 \text{ kHz}$  and measured with a grid of 2255 points ( $55 \times 41$ ).



(a) Bottom plate surface with displayed damages containing various sizes and shapes.



(b) Top plate surface with no damages (the measured surface).

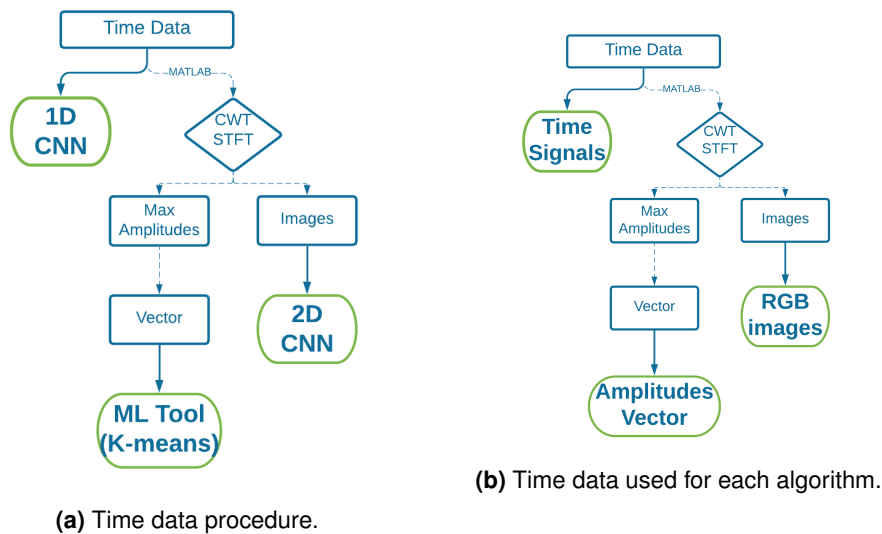
**Figure 6.2:** PMMA plate, from whose measurements will be used to present the algorithms' results.



# 7

## Time Data Procedure

This chapter contains the description of the algorithms related to the time data and also a brief description of the time data itself. Three algorithms were developed, two with Deep Learning (DL) techniques and one with Machine Learning (ML). Figure 7.1a shows the time domain procedure, and, as can be seen, one branch of algorithms was made to analyse processed data with two transformation techniques: Continuous Wavelet Transform (CWT) and Short-Time Fourier Transform (STFT). Both these transformations were used to create two different sets of data: amplitude vectors and images.



**Figure 7.1:** Overview of the methodologies developed for the time-domain analysis a) Time data procedure and b) input data for each of the corresponding algorithms.

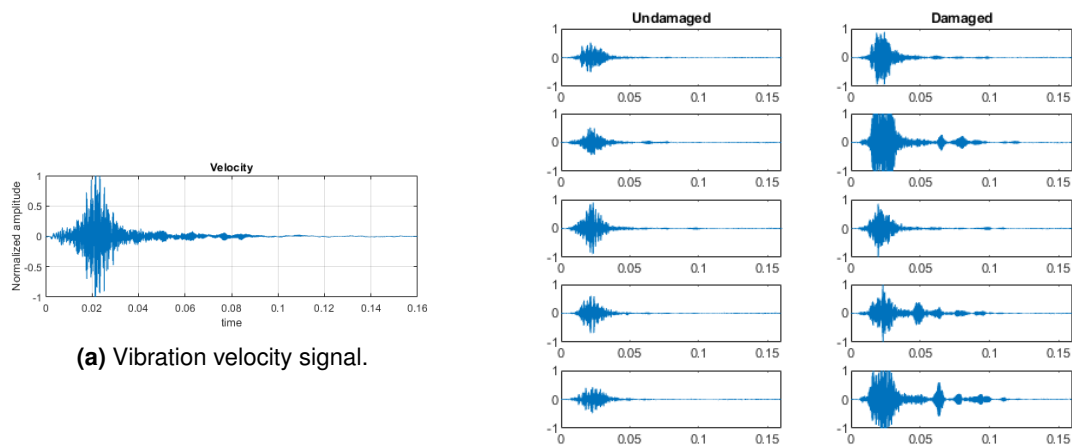
### 7.1 Time Data

There are different types of inputs to the algorithms (figure 7.1b) which are obtained from the Laser Doppler Vibrometer (LDV) and with the CWT and the STFT transformations. In this section, the raw time signals and the data obtained from these transformations and pre-processing procedures will be presented.

### 7.1.1 Time Signals

In a sequential order of steps after the LDV measurements, the time signals are the raw unedited, unprocessed data and thus the first step to be presented. The LDV measures for a grid of nodes, the vibration velocity obtained from the vibration of the plate subject to a PZT patch excitation. One example of a time-signal is shown in figure 7.2a, where the y-axis corresponds to the velocity amplitude, which has been normalized, and the x-axis corresponds to the time length of the measurement.

In the context of damage detection, a LDV measurement on a defected node will provide a different vibration signal than on a non-damaged node. In the context of Local Defect Resonance (LDR), this stands to reason since a damage will have higher vibration amplitudes. This phenomenon can be seen in the following figure 7.2b. On the right columns there are 5 time signals corresponding to defected nodes, and on the left column the same amount of signals on non-defected nodes. A 1D Convolutional Neural Network (CNN) was developed to analyse these time signals and recognise whether they correspond to a defected node or otherwise. For some of the following examples, the analysis may stand to a visual point of view, but in other cases, the damaged signal has a big resemblance to the non-damaged signal. Moreover, since the LDV measurements in this project can contain up to 10000 nodes, a visual inspection would be non-practical, and the 1D CNN can analyse this amount of signals in a matter of minutes.



**(a)** Vibration velocity signal. **(b)** Damaged and non-damaged velocity signals.

**Figure 7.2:** Time signals measured with the LDV on the PMMA plate.

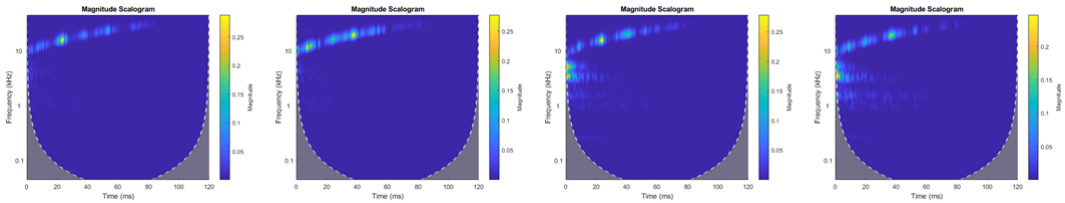
### 7.1.2 CWT and STFT Images

Regarding the time data, two time-frequency transformations were applied to the time signals. These are the Continuous Wavelet Transform (CWT) and the Short-Time Fourier Transform (STFT) which provide an analysis of time-varying frequency spectral characteristics, useful for defect detection purposes. Since the vibration of a defected node will be different than that of a non-defected node, both CWT and STFT can highlight this phenomenon. These transformations were applied to obtain two types of

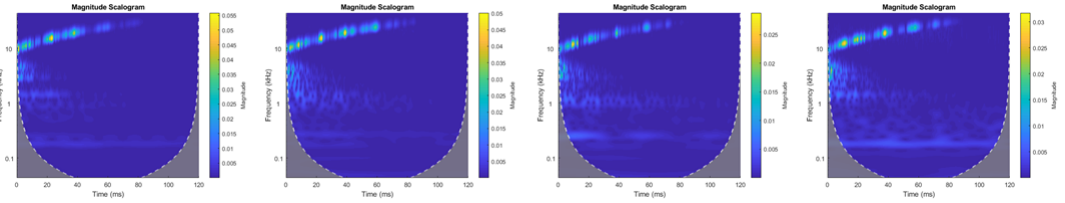
datasets: images of the transformation plotted on a frequency-time graph; and maximum amplitudes. In this section, the images will be explained and described.

**CWT Images**

Each of the following figures 7.3 and 7.4 contains the plot of 4 CWT applied to 4 time signals obtained from a measurement using the LDV to the PMMA plate, excited with a sweep-signal up to 40 kHz. The CWT is a frequency-time analysis and the plot of this transformation is consequently on a frequency-time scale (frequency is the y-axis and time the x-axis). The brightest the color on the picture, the highest the vibration amplitude at that corresponding frequency at that corresponding moment in the time signal. Moreover, the yellow-est point corresponds to where the vibration amplitude was maximum, therefore all images have a scaled color bar not representative of the same amplitude for the same color, across two different images.



**Figure 7.3:** Continuous Wavelet Transform (CWT) applied to 4 different **damaged** time signals.



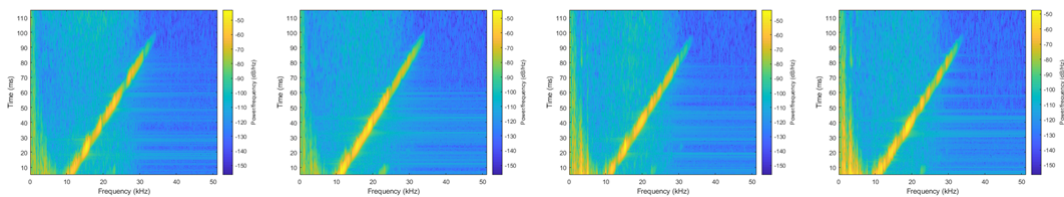
**Figure 7.4:** Continuous Wavelet Transform (CWT) applied to 4 different **non-damaged** time signals.

Observing all 8 sub-images, many resemblances can be observed. All images show a bright sweep which is indicative of the own excitation’s sweep up to 40 kHz. The differences from a damaged node CWT image and a non-damaged node CWT image is not obvious, since all images have a great similarity. However, the damaged images in 7.3 portray a blue-est plot, being that the ‘background’ of the images in Magnitude Scalogram in 7.4 is bluer (less bright). Since all the CWT are plotted with a scaled color code, the yellow-est point in two different images does not necessarily have the same amplitude and looking very closely into the scale of the damages CWT in comparison to the non-damaged CWT , there can be seen higher amplitudes on the color bar for the first case (damaged CWT have around 0,25 of maximum magnitude amplitude, whereas non-damaged CWT have around 0,05). Therefore, in terms of scaling, since the excitation can be considered approximately similar for all points and the amplitudes in the damaged images are higher, they will also be prone to show less bright colors correspondent to non significant amplitudes across the frequency-time spectrum. That is the reason behind the ‘clearer’ images for damaged time signals transformations in comparison to non-damaged time

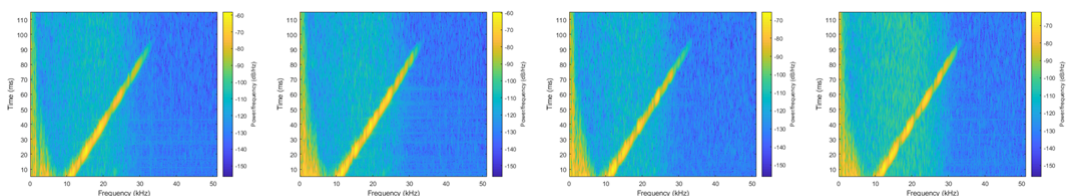
signals transformations, which meets the concept of Local Defect Resonance (LDR) (highest vibration amplitudes for the defect in comparison to the plate).

### STFT Images

Just as there could be observed some differences in the CWT frequency-time plots, the same can be expected from the STFT time-frequency plots. Each of the following figures 7.5 and 7.6 contains the plot of 4 Short-Time Fourier Transforms (STFTs). The STFT is a time-frequency analysis and the plot of this transformation is consequently on a time-frequency scale (time is the y-axis and frequency the x-axis). The STFT is a time-frequency analysis and the plot of this transformation is consequently on a time-frequency scale (time is the y-axis and frequency the x-axis). The brightest the color on the picture, the highest the vibration amplitude of that corresponding frequency at that corresponding moment. Comparing to the CWT plots, where the frequency axis was in a logarithmic scale, for the STFT plots, the frequency is in a linear scale. Moreover, similarly to the previous CWT images, the color bar is scaled, therefore it is not representative of the same amplitude for the same color, across two different images.



**Figure 7.5:** Short-Time Fourier Transform (STFT) applied to 4 different **damaged** time signals.



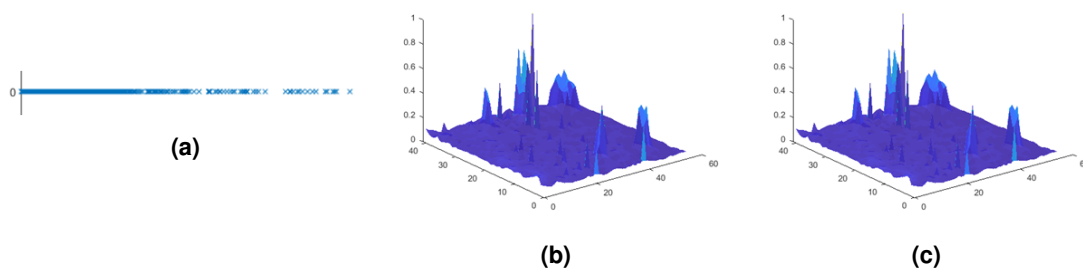
**Figure 7.6:** Short-Time Fourier Transform (STFT) applied to 4 different **non-damaged** time signals.

Observing all 8 sub-images, there can be observed many resemblances. All images portray a bright sweep which represents the excitation's sweep up to  $40\text{ kHz}$ . In a general notice, these images have higher resemblances than the CWT, therefore is less obvious to spot the differences from a defected and non-defected time signal transformation. Just like for the previously presented set of CWT images, identifying the difference from damaged and non-damaged STFT images lies as a direct consequence of the Local Defect Resonance (LDR) concept. For damaged nodes, the vibration amplitude will be higher and so will the magnitude of the transformation, which can be observed by a close inspection at the values in the color bars of each sub-figure. Whereas the non-damaged STFT have maximum power/frequency amplitudes of around  $-60\text{ dB}$ , the damaged have higher  $-50\text{ dB}$  maximum magnitudes. Therefore, in terms of scaling, since the excitation can be considered approximately similar for all points,

and the amplitudes in the damaged images higher, the images of damaged STFTs show less bright colors in the background, especially in the bottom right corner or each picture.

### 7.1.3 CWT and STFT Maximum Amplitudes

In this thesis project, these transformations were also applied in order to obtain the maximum amplitude of vibration for each time signal. That means registering the maximum amplitude corresponding to the magnitude of the brightest point of each CWT or STFT image, like the ones represented in 7.3 - 7.6. The result from registering the maximum amplitude point for each node of the measurement's grid will be a vector of amplitudes, with the size of the number of points in that grid. Therefore, the output of this operation will be a vector of points, like the one represented in figure 7.7a. This vector of amplitudes gains meaning when plotted in a grid of points matching the one used for the measurements. In the figures 7.7b and 7.7c this plot is presented respectively for the CWT and STFT vectors. Just by looking at these images, already some unusual agglomerations of high amplitude points can be observed. With further analysis using the algorithms developed in this thesis project, some defects will be identified using this input data.

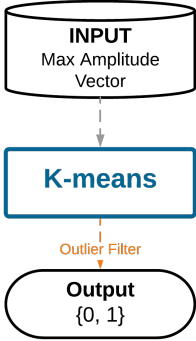


**Figure 7.7:** Result of the application of the CWT and STFT to the time-signals so to obtain the maximum vibration amplitudes. A vector of points like the one displayed in a) is obtained. Figures b) and c) show the plot of the respective CWT and STFT vector on a 2D grid with the dimension of the grid measured by the LDV.

## 7.2 Machine Learning Tool (K-means)

The first algorithm to be presented will be the ML tool that uses the K-means technique for amplitude clustering. The type of input given to this algorithm was described in sub-section 7.1.3. This is a rather simple algorithm in terms of stages, having a total of two operations (K-means and Outlier Filter). The most complex step was previously performed by the transformations. Meaningful results were obtained from this algorithm and they were the first step in this thesis project into having an algorithm capable of damage classification.

### 7.2.1 Methodology



**Input:** Vector containing for each node of the plate grid, the maximum amplitude obtained wither from the CWT or the STFT transformation (see section 7.1.3 for a characterization of the input);

**K-means:** Performs amplitude clustering separating the one dimensional vector of amplitude points into two clusters of points;

**Output:** Value 1 for all the points in the cluster with higher amplitudes, and value 0 for all the other points from the remaining cluster. Plotting the respective values in the respective plate grid, a defect map can be visualized;

**Outlier Filter:** Filters all the defect points detected alone in respect to their surrounding points by eliminating them as defects.

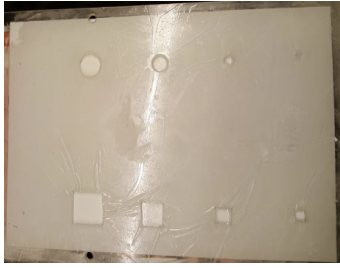
### K-means

As described in section 3.2, the K-means algorithm has some parameters that need to be described before its implementation. These are the number of clusters and iterations. Choosing the number of clusters depends on the implementation purpose. Since in this thesis project, applying the K-means technique to the vector of amplitudes aims to detect a cluster of defected points and other of non-defected, the number of clusters was chosen to be 2. As for the number of iterations, it should be chosen as such to imply convergence of the clustering process, and therefore, after some values were hand-tested, it was chosen to be 10. A visualization of the K-means algorithm running in a vector of amplitudes, with two clusters can be seen in figure 10.7b, in the simulation part of this thesis project.

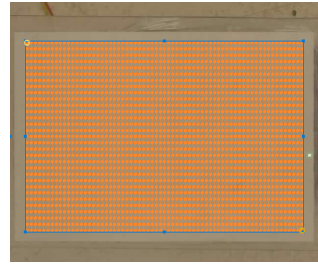
### 7.2.2 Results

After all steps of the ML tool (K-means) applied to the max amplitude vector obtained either from the CWT or STFT, the result will be a map with the dimensions of the LDV measured grid, containing either the value '0' (non-defected nodes represented in blue) or '1' (defected nodes represented in yellow). Hence the results are binary maps and are shown in figures 7.9 and 7.10.

Across all algorithms to be presented in part II (Defect Detection for Experimental Data), for comparative motives, the same dataset was analysed by all algorithms both for the Chapter 7 Time Data Procedure and the Chapter 8 Frequency Data Procedure. This dataset is constituted of 2255 measurement points distributed in a grid of 55 per 41 points equally spaced in the vertical and horizontal directions. In figure 7.8 such a grid of points is displayed. It is defined in such a way to measure the top surface of all eight defects with varying sizes and two different shapes. Therefore, in terms of detection objective, 8 defects would be the perfect result, although challenging since there are defects with a small size.



(a) Plate's bottom surface with 8 Flat Bottom Holes (FBHs) with varying sizes.

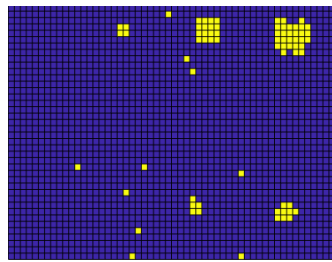


(b) Plate's top surface with the grid of points measured by the LDV at display.

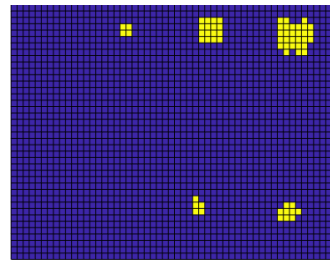
**Figure 7.8:** Measured PMMA plate correspondent to the results to be presented in part II (Defect Detection for Experimental Data).

### CWT results

Figure 7.9 displays the result of the ML tool (K-means) applied on the CWT maximum amplitudes. There can be seen that 5 out of 8 defects were detected. Also, some information of the squared defect shapes could be retrieved. Since an outlier filter is applied, the resolution of this method is 2 nodes per defect.



(a) Output before the outlier filter.

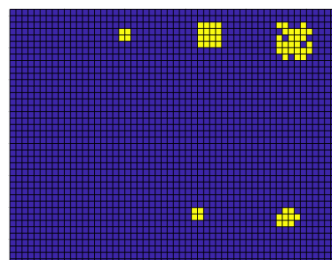


(b) Output after the outlier filter.

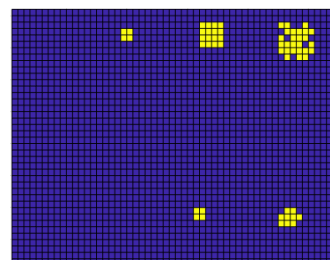
**Figure 7.9:** ML tool (K-means) applied on the CWT maximum amplitudes result.

### STFT results

Figure 7.10 displays the result of the ML tool applied to the STFT maximum amplitudes. Similarly to the result for the CWT, 5 out of 8 defects were identified. But, unlike the previous results, the outlier filter performed no operation, therefore one can also conclude that the STFT operations provides a less noisy dataset.



(a) Output before the outlier filter for the STFT results.



(b) Output after the outlier filter for the STFT results.

**Figure 7.10:** ML tool (K-means) applied on the STFT maximum amplitudes result.

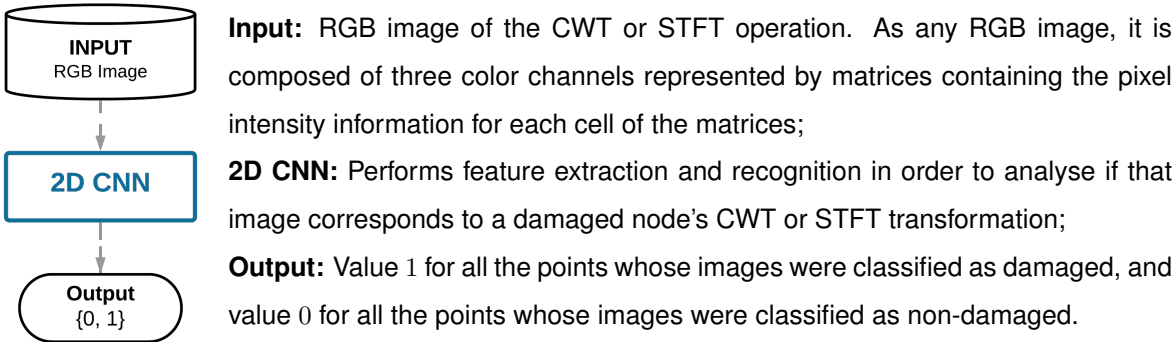
Summing all, the ML tool for the STFT and CWT maximum amplitudes provides the first approximation of a defect map, with the correct identification of 5 out of 8 defects. Moreover, one additional note can be stated, which is, since some of the algorithms presented in this thesis project are supervised learning algorithms, the labels for them to train were retrieved from these results of the unsupervised learning ML tool in the time domain. Therefore the defect maps shown in this sub-section will be relevant for the other algorithms.

### 7.3 2D CNN (Convolutional Neural Network)

Whereas the previously presented algorithm contained Machine Learning (ML) techniques, the one presented in this section will be the first presented Deep Learning (DL) algorithm. In Chapter 4, an introduction on the working principles and theoretical notions of Convolutional Neural Networks (CNNs) is shown. This algorithm uses a CNN to analyse RGB (red green blue) images obtained with the CWT or STFT, as previously described in sub-section 7.1.2.

In terms of working principle, as a supervised learning algorithm, the 2D CNN need a step of training, in order for the neurons to obtain valid values that serve the classification purpose they are being trained to. Therefore, using this algorithm involves always two steps: training (longest) and testing. In the next subsection, the overall methodology of the 2D CNN algorithm will be explained.

#### 7.3.1 Methodology



#### 2D CNN

In this thesis project, no 2D CNNs were crafted from origin, instead previously designed architectures were used and adapted to this project's classification problem. The concept of re-using a previous CNN architecture is called Transfer Learning (TL). Nonetheless, some changes were performed in the last layers, more specifically, in the last fully connected layer and the output classification layer in order to accommodate the CNN's classification to the project's purpose, which is a necessary step of this TL methodology.

The following 2D CNN architectures were tested: GoogLeNet [14], DenseNet-201 [36], Inception-ResNet-v2 [37]. Taking into account the computational time, GoogLeNet is the least expensive of the



three and since its results didn't have a significant variance from the other more complex nets, it was the CNN architecture chosen to proceed in this project. On a further note about the training of the CNNs, the optimization algorithm chosen was the Stochastic Gradient Descent with Momentum.

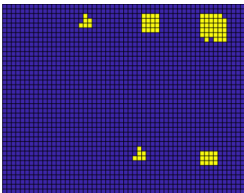


**Figure 7.11:** GoogLeNet architecture [14].

**7.3.2 Results**

Summarizing this algorithm's procedure, the LDV measures a grid of points as the one displayed in figure 7.8 and for each of the nodes (orange points) a time signal with the plate's response to the PZT patch excitation is registered. After, with the application of the CWT or the STFT to the time-signals, a time-frequency image representation is obtained for each node, which will be the input to the 2D CNN . After the classification of all images correspondent to the 2255 node dataset (55x41 grid of LDV measured points over the plate with eight defects), a defect map can be built from the '1' or '0' algorithm classification. The aim of the classification is to have defected node classifications around the defect locations.

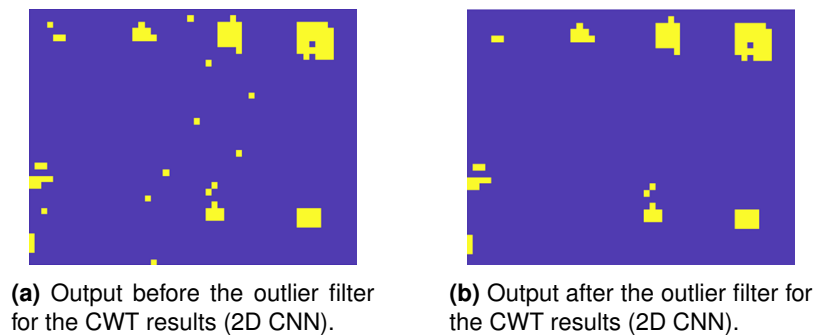
Moreover, as a supervised learning algorithm, the Convolutional Neural Networks (CNNs) require previously defined labels for the training procedure. However, since the measurements are being done looking at the top surface of the plate (which has no defects) with a grid containing many points, it is impractical and difficult to have a notion the ground truth. This is, since it's not visualized whether the measured nodes are on top of damages or not, it's not possible to visually label the signals for the further training of the algorithm. Thus, the output of the ML tool was used for labelling purposes to this 2D CNN and afterwards, for both time-domain and frequency-domain 1D CNNs. In this condition, fifty percent of the labels obtained in figure 7.12 were used to train these DL algorithms. Hence, in this project a ML algorithm (of unsupervised learning techniques) is used to train others (with supervised learning techniques), making possible to use powerful classification tools like CNNs to the damage detection purpose.



**Figure 7.12:** Label map for the CNNs

### CWT results

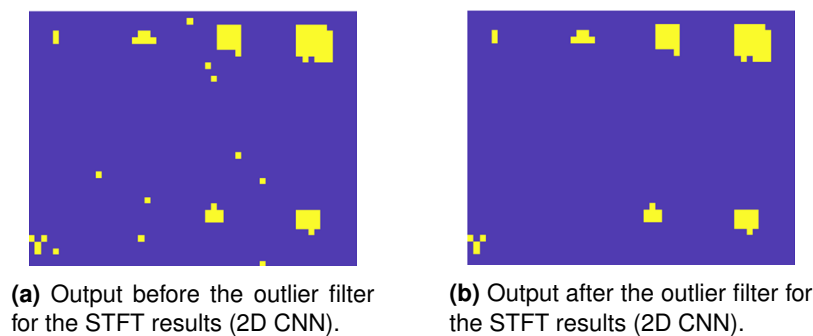
Figure 7.13 displays the 2D CNN results on the Continuous Wavelet Transform (CWT) images. Two images are displayed after and before the application of the outlier filter. Resultant from this application, a resolution of two nodes per defect is implied with the aim of removing points associated with noisy measurements, that affect the output results. Comparing to the results from the Machine Learning algorithm presented in figure 7.9, the 2D CNN result achieved the classification of one more defect (small rectangular defect on the top left corner). Moreover, looking at the bottom left corner of the images, a defected region is identified. This region corresponds to the location where the PZT patch was glued to the plate in order to provide the excitation. Thus, this algorithm detects also the presence of the PZT patch. On a final note, considering that some defects were also misclassified, the 2D CNN applied to the CWT images detects 6 out of 8 plate defects.



**Figure 7.13:** 2D CNN applied on the CWT time-frequency image representation result.

### STFT results

Figure 7.14 displays the 2D CNN results on the Short-Time Fourier Transform (STFT) images. Similarly to the results described above, two images are displayed which correspond to the results obtained before and after the application of the outlier filter. Comparing to the CWT results above, the STFT version had less misclassifications and thus a clearer image regarding the defect locations. The sixth defect (top left corner) and the PZT patch location (bottom left corner) were also detected, in an overall classification that provides a clear insight to the location and also some information regarding the shape of 6 out of 8 defects of the PMMA plate.

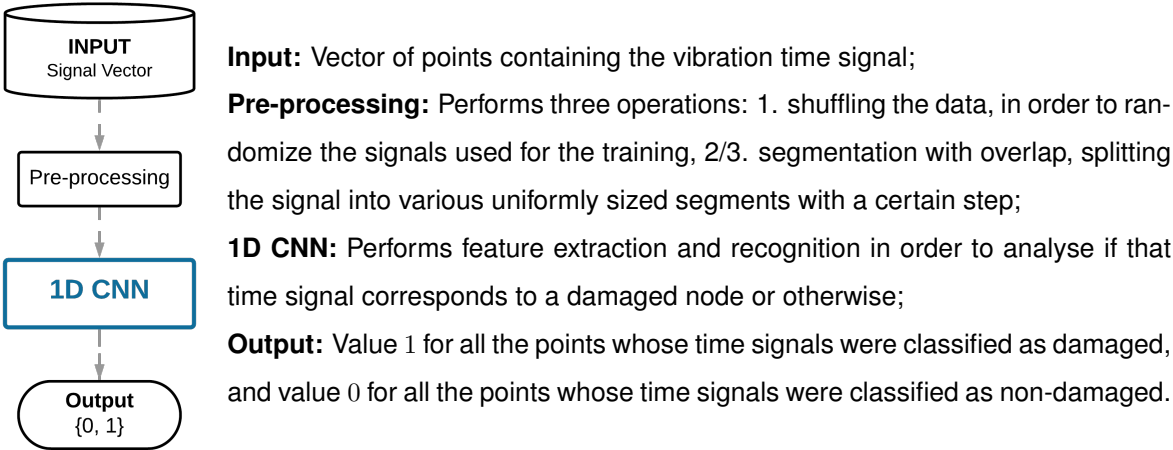


**Figure 7.14:** 2D CNN applied on the STFT time-frequency image representation result.

# 7.4 1D CNN (Convolutional Neural Network)

Observing figure 7.1a, there can be seen that the 1D CNN was made to analyse directly the raw time signals measured by the LDV. As described by section 7.1.1, the time signals are one-dimensional vectors containing the vibration time history of each measured node. Therefore, the most commonly used types of CNNs which contain 2D kernels to analyse images, cannot be used. Therefore, in order to analyse this one-dimensionally represented data, a CNN using 1D kernels was developed. Two-dimensional kernels contain the squared amount of neurons than a one-dimensional kernel, thus it is important to mention that this algorithm does not involve as big an optimization problem as for the previously presented 2D CNN .

## 7.4.1 Methodology



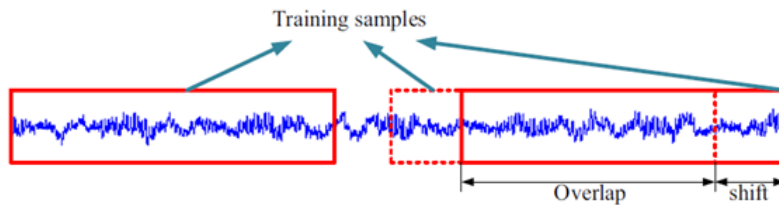
### Pre-processing

This step from this algorithm’s methodology was critical for the success of its implementation. One of the main reasons is that a time signal can contain up to 12k samples (points). This number is too high for a CNN to properly perform feature extraction and recognition by analysing the entirety of the signal directly. That was the first motivation for designing this step. This is, segmenting the signal into equally sized segments containing a small portion of the time signal. Moreover, by applying this segmentation procedure, another advantage can be obtained. By truncating the signal into equally sized segments, but selecting the next segment to include still a part of the previous one, much more data can be obtained. This is, by performing this segmentation procedure with overlap of segments, much more than 12 segments of 1k samples can be extracted from a time signal containing 12k samples. Since the amount of data is crucial for the training of DL algorithms, implementing this overlap procedure allowed to obtain better accuracies.

One other operation is performed in this pre-processing step, the data shuffling. This means that, from a certain dataset, containing a certain amount of time signals from defected nodes and other amount from non-defected nodes, by mixing the data before it is given for the network to train, one

can remove some biasing from the overall procedure. The previously described operations can be sequentially presented as:

1. **Shuffling:** Randomly mixing the time signals in order not to have biased CNN trainings;
2. **Segmentation:** Splitting the time signal into equally sized segments;
3. **Overlap:** Introducing a certain shift for the segmentation procedure to generate more segments and therefore data from an original time signal.



**Figure 7.15:** Overlap operation.

## 1D CNN

Contrary to the use of 2D CNNs, the use of 1D CNNs involved crafting from origin the network. This means designing the architecture in terms of convolutional, pooling, batch normalization, fully connected and the remaining types of layers including their parameters such as kernel sizes and amount of filters. In table A.1 of the appendix A, a full description of the 1D CNN architecture is presented, along with the kernel sizes, the layer operations and so on. The architecture can be summed up to 5 convolution, batch normalization, ReLU, pooling blocks, followed by two fully connected layers with one dropout layer, ending with a softmax and classification layer. The number of kernels has a tendency to increase by a factor of 2 across the network until reaching a value of 64. On a further note, the use of a first convolutional kernel much bigger than the ones in deeper convolutional layers is done with the aim of suppressing the high-frequency noise. The process to obtain such an architecture was trial and error, where parameters such as the kernel sizes and blocks were studied from a comparison of the corresponding obtained results.

### 7.4.2 Results

Figure 7.16 displays the 1D CNN results on the velocity vibration time signals. The defect map shown in that image was built from the classification of all time-signals measured with the LDV into defected and non-defected. It is the same dataset as the one evaluated from the ML tool and the 2D CNN and from a comparative point of view, one more defect was detected (the third circular defect). Therefore, 7 out of 8 defects have been detected by this method, missing only the smallest circular defect. Increasing the number of points on the grid and therefore the number of points measured on top of the defects, associated with increasing the excitation frequency (this dataset was obtained with a 0 - 40 kHz excitation

band) would increase the chances of detecting this last defect. Moreover, the PZT patch was also detected as can be seen by looking at the bottom left part of the map.



**Figure 7.16:** 1D CNN for time-domain data result.

Considering the training of the 1D CNN , similarly to the 2D CNN , the labels were retrieved from fifty percent of the ML tool's results. In sub-section 7.3.2, a thorough description of this reasoning is written. The map used to train contained 5 defect locations and the 1D CNN found two more plus the location of the PZT patch. This is an evidence of this algorithm's ability for feature learning and feature extraction.

### 7.4.3 Bayesian Optimization

As explained in the theoretical part, dealing with DL techniques involves dealing with many parameters at the training level, like the learning rate and learning rate drop factor, and also at the architecture level, such as defining the kernels of the convolutional and pooling layers along with the number of layers itself. Moreover, obtaining the best performance and the most accurate results from such algorithms implies obtaining an optimized solution for these parameters. For such a case, an accurate feature extraction and classification made by the CNN would be obtained, as a consequence of a reliable training with tuned parameters for the back-propagation procedure.

For this purpose, the Bayesian Optimization was implemented to the time and frequency-domain 1D CNN in order to obtain the best as possible architecture with the best training hyperparameters for the used back-propagation algorithm (Stochastic Gradient Descent with Momentum). It is also important to notice that the previous 1D CNN architecture presented for this algorithm was one designed by hand, with a test and trial procedure to obtain a reliable solution, capable of making good classifications.

Table 7.1 illustrates the variability of values that these parameters can have. Four parameters regarding the training are listed, which are related to the Stochastic Gradient Descent with Momentum back-propagation algorithm. As can be seen, the corresponding intervals have a big variance of values and obtaining a tuned solution for them without an optimization algorithm requires a process of extensive test and trial or great previous knowledge on the working principles of such an algorithm and the problem at hand. Hence, these four hyperparameters were left for the Bayesian Optimization, aiming to obtain the best set of values capable of training the neurons of the 1D CNN into tuned values for the task of damage detection.

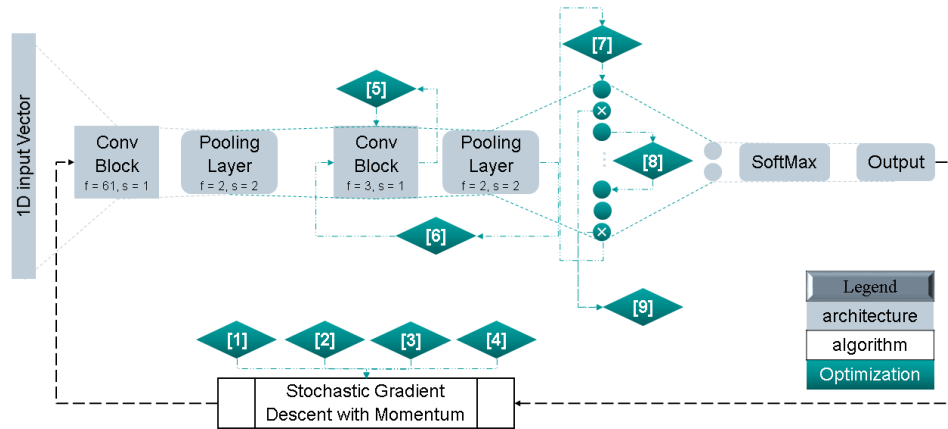
Other than the training parameters, also the architecture parameters were optimized aiming to get the best as possible architecture for the analysis of the time-signals. Although already one successful 1D CNN architecture was found, whose results were presented in the previous sub-section, this thesis

**Table 7.1:** Parameters left for Bayesian Optimization regarding the training and architecture of the 1D CNN

Training Parameters		Architecture Parameters	
Learning rate [1]	[2e-4, 8e-2]	Convolutional blocks [5]	{1, 2, 3}
Learning rate drop factor [2]	[0.05, 0.5]	Conv/pool blocks [6]	{2, 3, 4, 5, 6}
Stochastic gradient descent momentum [3]	[0.8, 0.98]	Fully connected layers [7]	{1, 2}
L2 regularization parameter [4]	[1e-10, 1e-2]	Neurons in the fully connected layers [8]	{200, 300, 400, 500}
		Dropout factor [9]	[0.2, 0.8]

project aimed at obtaining another architecture resultant from this optimization procedure. Some similarities were maintained, such as the convolutional block with a big convolutional kernel and obviously the last layers dedicated to the classification problem. As can be seen in table 7.1, the number of convolutional layers before a pooling layer, the number of convolutional/pooling layer blocks, the number of fully connected layers and their number of neurons were left to optimization. Moreover, also the dropout factor was optimized.

It's important to mention that in order to optimize so many parameters, related to different aspects of the 1D CNN, this optimization ran for around two hundred cases, building a total of 16 hours. The interval of values shown in the table above was the interval of values given for the optimization. Figure 7.17 shows the graphic representation of the architecture's elasticity being created with this methodology. Moreover, it shows a sum-up of the different optimized parameters, along with their place in the 1D CNN. The code of numbers is also correspondent to the one in the table 7.1.



**Figure 7.17:** Optimization methodology representation.

The final result of the Bayesian Optimization will be a set of tuned parameters based on the best performance in the task of damage classification. Table 7.2 shows the results for the training hyperparameters. Regarding the optimized architecture, it is shown in table A.2 of the appendix A.

**Table 7.2:** Optimized training hyperparameters for the time-domain 1D CNN

Training Hyperparameters	Optimized Values
Learning rate	0.0007
Learning rate drop factor	0.4995
Stochastic gradient descent momentum	0.8744
L2 regularization parameter	7.2038e-9

From a comparison between the optimized CNN (table A.2) and the hand-made CNN (table A.1)

described previously in this section, there can be seen that the optimized architecture has less convolutional layers and more fully connected layers, for a total of 4 blocks (previously there were two). This can be seen as not having such a high need for the feature extraction operations done in the convolutional layers and needing extra neurons for the feature classification part of the 1D CNN , by having more fully connected layers.

Across the presentation of the different algorithms in this chapter, one equal dataset was used for all algorithms to run, in order to have coherence in the comparison of results. However, in order to test better the architecture obtained with the Bayesian Optimization, a bigger dataset was used, which contains around 3700 points (previous dataset contained around 2300). The labels which served to train the CNN and the result from the optimized CNN follow in the next pictures. Equally to the use of the other CNNs in this thesis project, the optimized CNN trains with 50% of the labels and performs classification on the entire dataset.



**Figure 7.18:** Figure a) represents the results obtained with the optimized 1D CNN on the PMMA plate, which was obtained from the Bayesian Optimization in relation to the labels in figure b).

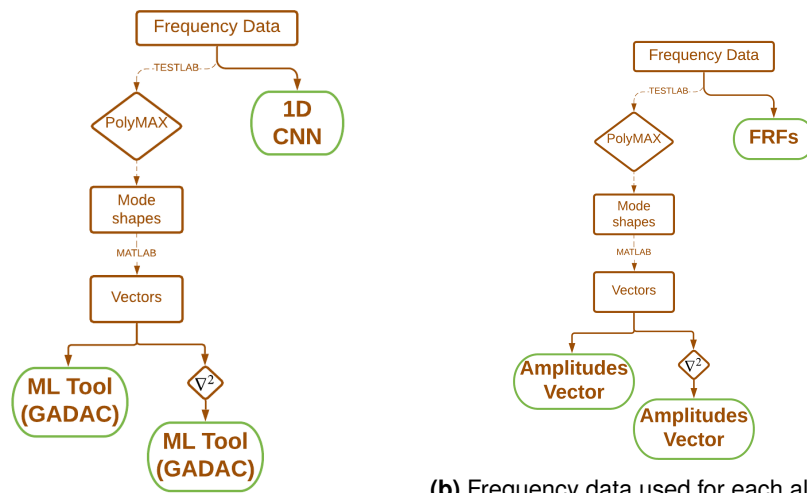
The Bayesian Optimization based itself on finding the best CNN architecture in relation to the defect map shown in figure 7.18b. That defect map contains seven defects out of eight, obtained by manipulating the results of the ML tool algorithm classification on the same dataset. Once again, the use of labels obtained from an unsupervised algorithm of this project is justified by the fact that no tangible method was found to be used in order to retrieve the ground truth (exactly which of the LDV measured nodes are on top of defects, or not).

Looking at the classification result, from the 1D optimized CNN , 7 defects were accurately identified along with the piezopatch location, in the middle of the plate. Thus, there can be concluded that the Bayesian Optimization results, in terms of architecture and training parameters provided meaningful results and a CNN capable of accurately classifying the defects locations.

# 8

## Frequency Data Procedure

This chapter contains the description of the algorithms related to the frequency data and also a brief description of the frequency data itself. Two algorithms were developed, one with Machine Learning (ML) techniques and one with Deep Learning (DL). Figure 8.1a shows the frequency domain procedure, and, as can be seen, one branch of algorithms was made to analyse processed data with Siemens Simcenter TestLab® using the Polymax tool [38]. From this process the mode shapes could be calculated, and were further input into the ML algorithm.



(a) Frequency data procedure.

(b) Frequency data used for each algorithm.

**Figure 8.1:** Overview of the methodologies developed for the frequency-domain analysis a) Frequency data procedure and b) input data for each of the corresponding algorithms.

### 8.1 Frequency Data

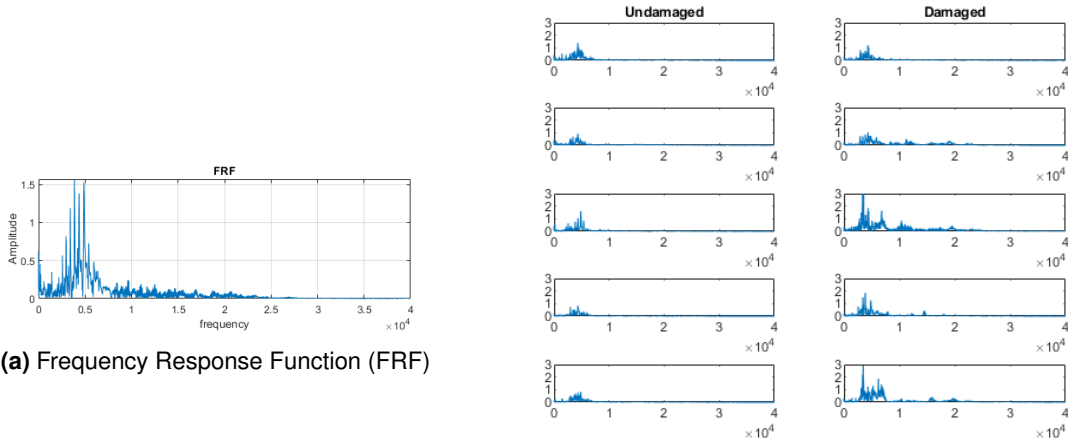
There are different types of inputs to the algorithms (figure 8.1b) which are obtained from the Laser Doppler Vibrometer (LDV), and with modal analysis using Siemens Simcenter TestLab® to obtain mode shapes. In this section, the Frequency Response Functions (FRFs) and the data obtained from modal analysis will be presented.



### 8.1.1 Frequency Response Functions (FRFs)

In a sequential order of steps after the LDV measurements, the Frequency Response Functions (FRFs) are the raw unedited, unprocessed data and thus the first step to be presented. The LDV measures for a grid of nodes, the FRF obtained from the vibration of the plate subject to a PZT patch excitation. One example of a FRF can be seen in the figure 8.2a.

In the context of damage detection, a LDV measurement on a defected node will provide a different FRF in comparison to a non-damaged node. This phenomenon can be seen as a consequence of the Local Defect Resonance (LDR) concept, since a damage will have higher vibration amplitudes. Figure 8.2b shows 5 FRFs measured on defected nodes and 5 other FRFs measured on non-defected nodes, where the difference above described can be seen. A 1D Convolutional Neural Network (CNN) was developed to analyse these FRFs and recognise whether they correspond to defected nodes or otherwise. Some differences stand to a visual point of view, but in other cases the damaged FRFs have a big resemblance to the non-damaged FRFs. Moreover, since a big number of FRFs are measured with the LDV, the use of DL algorithms provides the advantage of analysing all automatically in a matter of minutes.



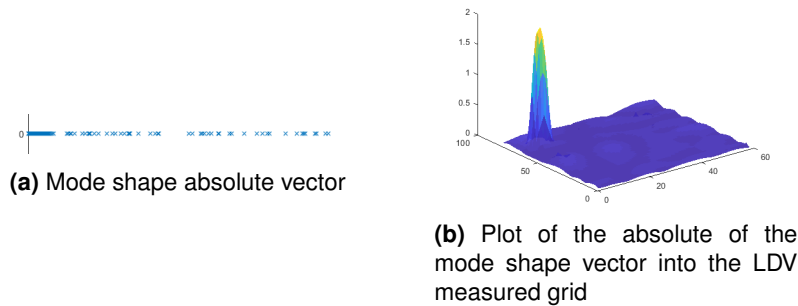
(b) Damaged and non-damaged Frequency Response Functions (FRFs).

Figure 8.2: FRFs measured with the LDV on the PMMA plate.

### 8.1.2 Mode shapes

By importing the FRFs into Siemens Simcenter TestLab<sup>®</sup>, modal analysis was performed in order to obtain the mode shapes. Within the software, the Polymax tool was used. This tool applies the Least Squares Complex Frequency Domain method (LSCF) in order to obtain the modal parameters (natural frequencies, damping factors and mode shapes). Therefore, the outcome of this procedure are multiple calculations of mode shapes for sequential frequency bands, which are represented by vectors of complex amplitudes (one point per measured node). Figure 8.3a shows the absolute values such a vector and the correspondent mode shape. Looking at the vector, many points can be observed with

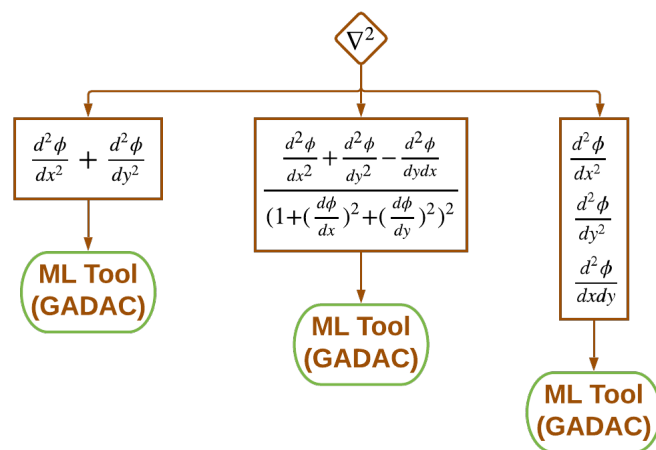
close-to-zero amplitudes and a smaller amount of points with distinctively high amplitudes. Looking at the mode shape plot, those bigger amplitudes can be identified as belonging to the Local Defect Resonance (LDR).



**Figure 8.3:** (a) Mode shape absolute vector. (b) Plot of the absolute of the mode shape vector into the LDV measured grid

### 8.1.3 2nd Gradients of mode shapes

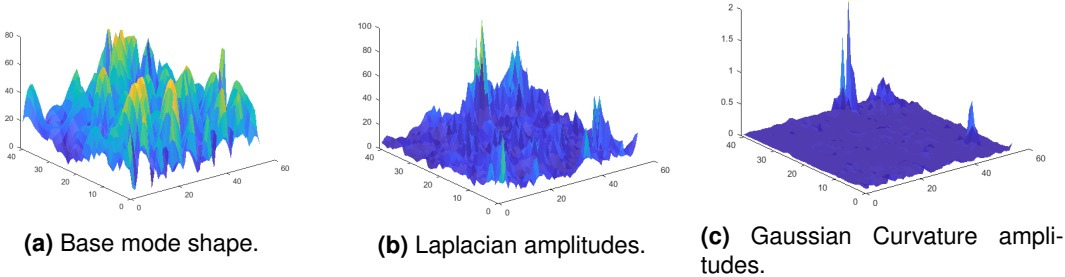
In order to better highlight the presence of a defect in a mode shape, in this project, some procedures of applying second gradients to the matrix of complex amplitudes were performed. For each 2nd Gradient method, there is a correspondent version of an algorithm as can be seen looking at the longest algorithm branch in figure 8.1a. After the application of the 2nd Gradient, another complex amplitudes vector is obtained, just as the one shown in figure 8.3a. Of these, there can be distinguished in figure 8.4: the Laplacian (procedure to the left), the Gaussian Curvature (procedure to the middle), and the 2nd Derivatives (procedure to the right). For each 2nd Gradient technique there is a version of the Machine Learning tool, in the overall methodology.



**Figure 8.4:** 2nd gradient procedures.

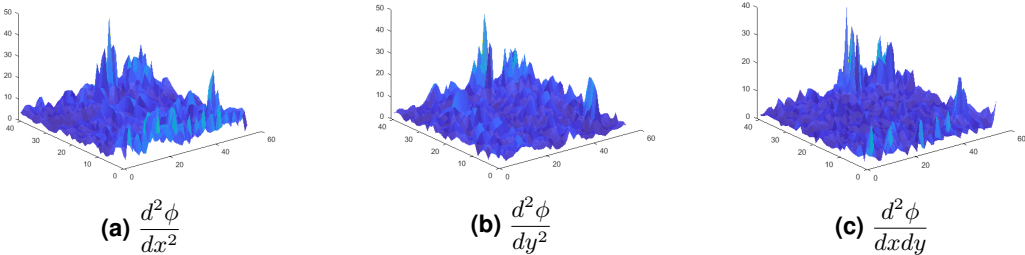
Figure 8.5 shows the effects of applying the Laplacian (figure 8.5b) and the Gaussian Curvature (8.5c) techniques to a reference mode shape (figure 8.5a). The Laplacian achieved some success in highlight the defect's presence, especially for the biggest defects. The Gaussian Curvature achieved

further success in highlighting three regions of high amplitudes corresponding to the location of three defects in the PMMA plate. However, as will be seen later, this doesn't mean that this will be the best 2nd Gradient version of the algorithm, since there are many mode shapes to be analysed in the 40 kHz excitation band provided to the plate.



**Figure 8.5:** Comparison of a) reference mode shape and the result of applying the b) Laplacian and c) Gaussian Curvature 2nd Gradient procedures.

Figure 8.6 shows how the 2nd Derivatives can highlight a defect's presence, but not necessarily better than the previous two versions. What distinguishes this 2nd Gradient technique in comparison to the others is that all the derivative calculations will separately be analysed by the algorithm, whereas in the previous cases, only the final result of the calculation will be submitted into the algorithm. This is the reason behind displaying the results of each of the independent derivatives.



**Figure 8.6:** Plot of the 2nd Derivatives amplitudes in relation to the base mode shape ( $\phi$ ) in Figure 8.5a

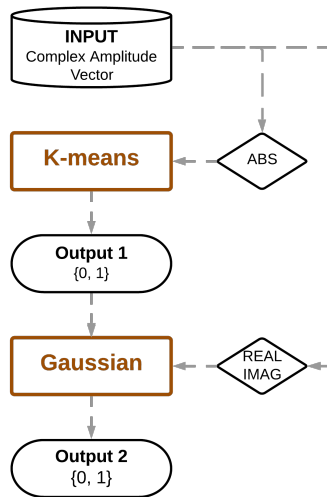
## 8.2 Machine Learning Tool (GADAC)

The first algorithm to be presented for the frequency domain procedure will be the ML tool that uses the so baptised Gaussian Anomaly Detection Automated by Clustering (GADAC) technique. This technique is a combination of the K-means clustering technique and the Anomaly Detection based on Multivariate Gaussian Distribution presented in the theoretical Chapter 3 Machine Learning. Comparing to the ML tool presented for the time domain procedure (section 7.2) this algorithm is a step further in terms of complexity, since it integrates one more technique.

This algorithm has as input the mode shapes or the 2nd Gradient of the mode shapes, as described previously in subsections 8.1.2 and 8.1.3. The modal analysis in the 40 kHz frequency band results in the determination of a big amount of mode shapes (around 200). Thus, this algorithm provides a fast tool to analyse hundreds of mode shapes, or their 2nd Gradients, obtained for high frequencies, where

the Local Defect Resonance dictates that there is localized damage behaviour.

## 8.2.1 Methodology



**Input:** Vector of complex amplitude values containing the mode shape or its 2nd Gradient information (see subsections 8.1.2 and 8.1.3 for a characterization of the input);

**K-means:** Performs amplitude clustering separating the one dimensional vector of absolute amplitude points into three clusters of points;

**Output 1:** Value 1 for all the points in the cluster with higher amplitudes, and value 0 for all the other points from the remaining clusters. Plotting the respective values in the respective plate grid, a defect map can be visualized;

**Gaussian:** Fits a Multivariate Gaussian distribution to the real and imaginary parts of the amplitudes dataset. An optimal threshold to separate defected from non-defected nodes is obtained using as a reference the defect map obtained from the previous K-means step;

**Output 1:** Value 1 for all the points below the optimal threshold, and value 0 for all the other points above the threshold. Plotting the respective values in the respective plate grid, a defect map can be visualized.

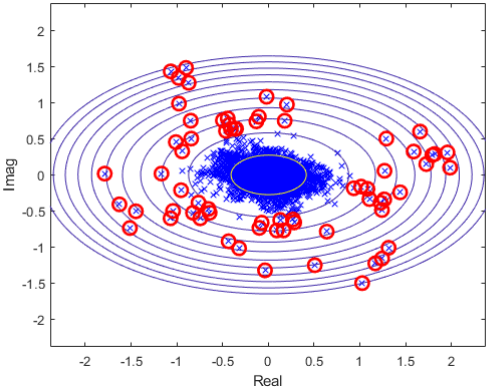
### K-means

Regarding the specifications of the K-means application to this algorithm, the number of clusters was chosen to be 3. This choice as to a more obvious one of selecting 2 (one cluster of defected and other of non-defected) was due to the fact that the many mode shapes analysed by the algorithm are combined modes. Therefore, 3 cluster do a better job to separate the vibration of the defect nodes from the one of the non-defected nodes for both the mode shapes and their 2nd Gradients. Hence, this same reasoning was applied to the analysis of the Laplacian and the 2nd Derivatives versions of the 2nd Gradient techniques. But for the Gaussian Curvature input data, since it is a powerful tool to highlight the defect's presence, the number of cluster was chosen to be 2, only for this input data. A visual representation of the K-means algorithm running on an amplitude vector with 3 and 2 clusters can be seen in figures 10.7a and 10.7b, respectively. Moreover, after a trial process to select the number of iterations needed for the algorithm to converge, 50 iterations were used.

### Gaussian

After a first defect map is obtained from the K-means clustering process, this map will serve as a reference for a threshold selection process to the Anomaly Detection based on Multivariate Gaussian Distribution algorithm. This threshold selection process is described in theoretical section about this algorithm (section 3.3). Having the threshold chosen, the training examples consisting of the real and

imaginary values of the complex amplitude vector will be separated into damaged and non-damaged. A visual representation of this process can be seen in figure 8.7.

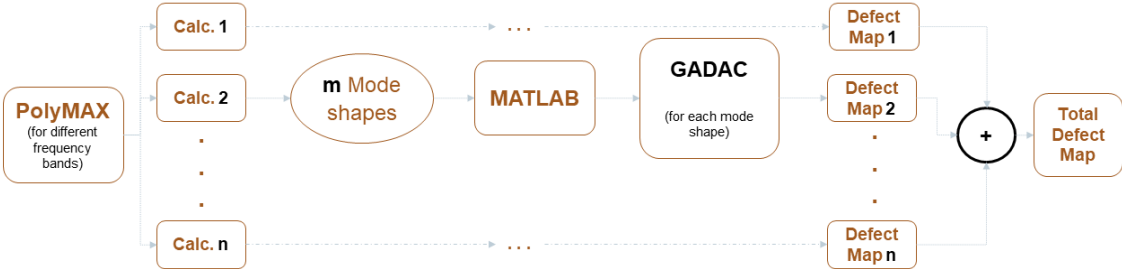


**Figure 8.7:** Anomaly detection based in Multivariate Gaussian Distribution representation.

**8.2.2 Results for the mode shapes**

The algorithm methodology described previously is applied sequentially to all mode shapes in the 0 to 40 kHz. After the ML tool (GADAC) analyses one mode shape, a defect map will be obtained and will be summed to the defect maps previously obtained for previously analysed mode shapes. The defect detection results of this algorithm will be as better as the more times the defected regions are detected from the analysis of multiple mode shapes. This described methodology is represented in figure 8.8. Information regarding the application of this algorithm independently to one mode shape can be found in the appendix section B.1.

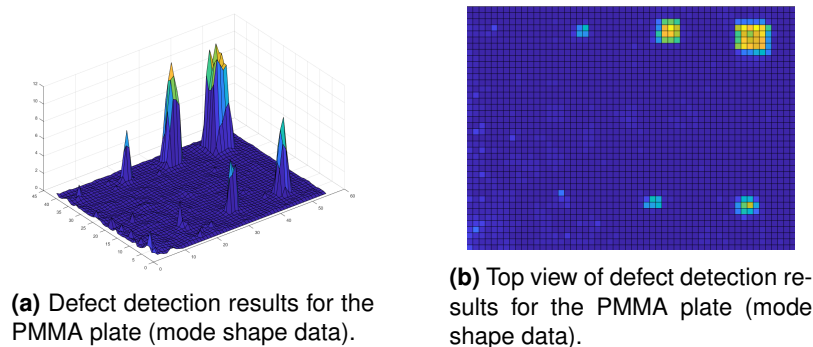
Using Polymax, the mode shapes are calculated for a selected frequency band. In the context of this project, since the total excitation included a band of 40 kHz, these frequency bands were usually considered to be 1 – 3 kHz to cope with the software’s capability to calculate mode shapes from the high amount of measured FRFs. Consequently, multiple calculations containing multiple mode shapes were obtained, which were all analysed sequentially and their defect maps summed, to originate the total defect map.



**Figure 8.8:** ML tool implementation methodology related to the Polymax mode shape calculation.

After this methodology is completed, the result will be the total defect map displayed in figure 8.9. Two images are displayed in this figure, one for the top view of the detection results and other for the

perspective view which is helpful to retrieve information regarding the results. The z-axis of the image represents a measure of number of calculations where the defect was highly identified in, thus it has no physical meaning, it is only a detectability measure. Regarding these results, the visual inspection is the important step to take conclusions for the position and shape of the defects. This algorithm accurately classified 6 defects (3 squared and 3 circular) along with a slight detection of the smallest squared defect. The shapes of the squared FBHs are well identified the circular FBHs are defined with non squared shapes (but also not perfect circular shapes). Additionally, the ML tool also detected irregularities in the bottom left corner of the plate, where the PZT patch was attached to provide the excitation.



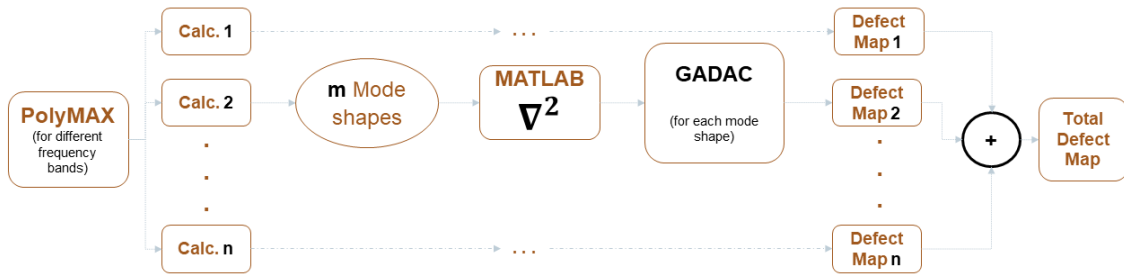
**Figure 8.9:** ML tool (GADAC) results for the mode shape data.

Comparing to the ML tool of the time-domain procedure, which analysed the CWT or STFT maximum amplitudes, the results for this frequency-domain ML tool provide clearer results with the detection of at least one more defect. It is also important to mention that, comparing to the time-domain ML tool, this algorithm doesn't run with an outlier filter and thus its resolution can be one node per defect. In the end, the results are also less noisy.

### 8.2.3 Results for the 2nd Gradients of the mode shapes

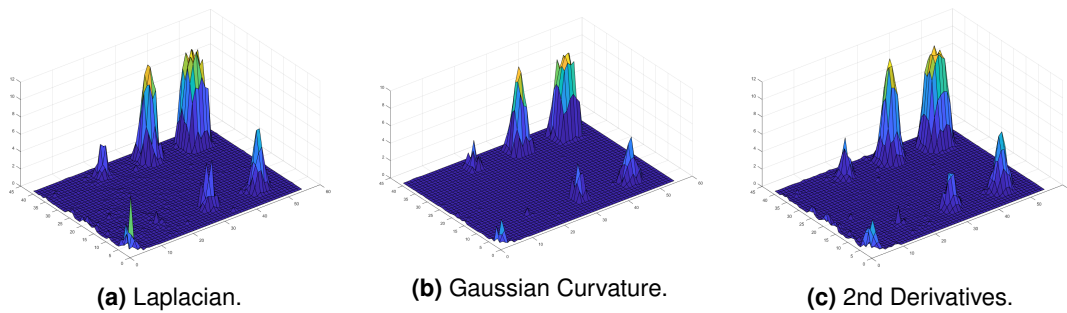
The ML tool (GADAC) method of application to the 2nd Gradient of the mode shapes, is equal to the method of application to the mode shapes. The difference lies in the pre-processing data step of calculating the 2nd Gradient after the mode shapes were obtained from Polymax, as shown in figure 8.10. In the last step of the methodology, equally a total defect map is calculated, from the analysis of the multiple mode shape 2nd Gradients calculated for each of the mode shapes within a Polymax calculation for a certain frequency band. Information regarding the application of this algorithm independently to one 2nd Gradient of a mode shape can be found in the appendix section B.1.

Considering the calculation of the 2nd Gradients, as explained in sub-section 8.2.2, three different techniques were applied: the Laplacian, the Gaussian Curvature and the 2nd Derivatives. Therefore there will be three results, which relate to the analysis of the same dataset measurement on the PMMA plate (2255 nodes) used to present the results for each of the previous algorithms, including the ML tool (GADAC) applied to the mode shape. Figures 8.11 and 8.12 show the results from the total defect maps obtained with each of the 2nd Gradients, in a perspective and top view, respectively.

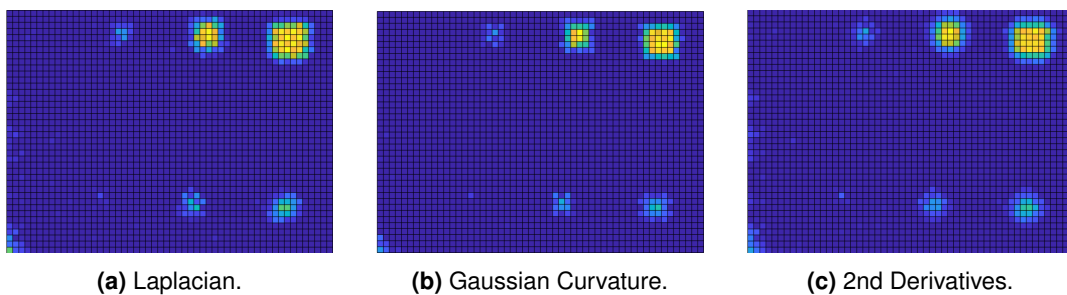


**Figure 8.10:** ML tool implementation methodology related to the Polymax mode shape calculation and the 2nd Gradient application.

For each of the 2nd Gradient versions there are two sub-figures, which are useful to be analysed in order to conclude about the defect detectability outcome. The meaning of the z-axis and the colors is related to the number of calculations a node was detected as defected. The highest the value, the highest the detectability, and the brightest the color. On a quick analysis of the results, the detection of 6 defects plus the PZT patch's location in the bottom left corner was achieved.



**Figure 8.11:** ML tool (GADAC) - perspective view of the total defect maps obtained for each of the 2nd Gradients.



**Figure 8.12:** ML tool (GADAC) - top view of the total defect maps obtained for each of the 2nd Gradients.

Although all three results look fairly similar, hence the success of this algorithm's damage detection for the 2nd Gradients, some differences can be identified from taking a closer look into these images. Firstly a measure of the algorithm's noise can be associated with the detectability results for the PZT location. The Gaussian Curvature version provided the less detectability results for the bottom left corner where the PZT was attached to the plate, thus verifying its strong capability of retrieving only the defect's influence in the mode shapes and at the same time cancelling the other areas of the plate not affected by the defects. However, looking at the smallest defect detected, the third circular defect, the Gaussian Curvature is also the one which has less resolution on it. Hence there is a trade off of clearer results,

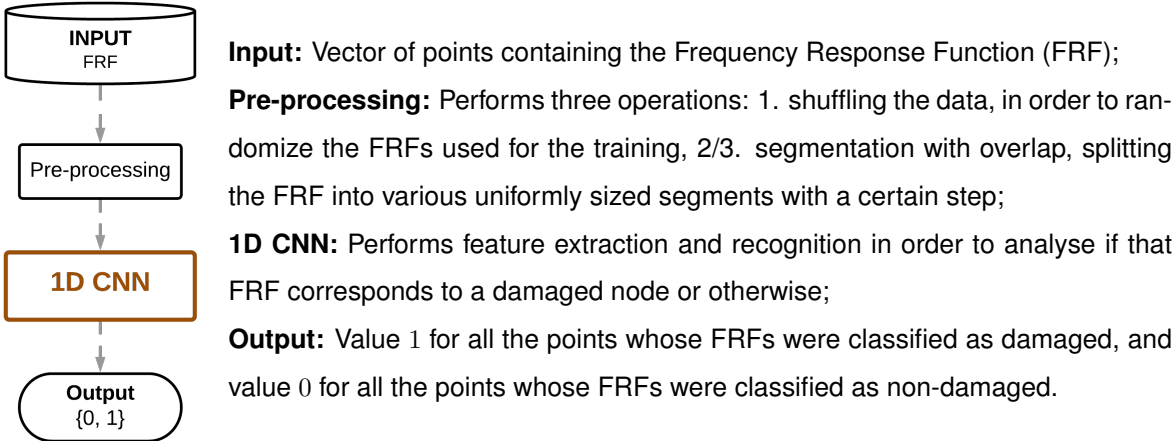
but with less resolution for the smaller ones. In this aspect, the 2nd Derivatives is the version that provides higher detectability of this smaller defect, and at the same time providing the clearest results regarding the shape of the defects. Regarding this matter, the squared defects have their shapes more or less defined by all versions, but the 2nd Derivative results shape the circular defects better than the others. One can conclude then, that the 2nd Derivatives version provides a good solution in terms of detectability for the smaller defects, at the same time captioning its shapes.

To conclude, from the PMMA plate containing 4 squared defects and 4 circular defects with decreasing sizes, three out of four circular and squared defects were detected by this algorithm. Moreover, the PZT patch's location in the bottom left corner was also detected. A note can be made regarding the fact that both defects missing detection are the ones with the smaller size. Therefore, measuring the plate with a thinner grid of points and increasing the excitation's frequency to higher values could provide for the detection of these smaller defects, by the algorithm. Nonetheless, this ML tool's results can be stated as a detection improvement regarding the results of the ML tool in the time-domain, since one more defect was detected.

### 8.3 1D CNN (Convolutional Neural Network)

The previously presented algorithm ran on data processed by Siemens Simcenter TestLab® . The 1D CNN runs on data directly obtained from the LDV , as can be seen by the scheme in figure 8.1b. The direct measured output frequency data are the FRFs, which are characterized by one-dimensional vectors containing the response behaviour of that particular measured node, across the excitation frequency band (subsection 8.1.1). Being one-dimensional data, the kernels from the Convolutional Neural Network (CNN) must also be one-dimensional, similarly to the previously presented 1D CNN for the Time Data Procedure, in section 7.4.

#### 8.3.1 Methodology





## Pre-processing

In both 1D CNN algorithms for Time Data and Frequency Data Procedures, the pre-processing step was critical for the success of their implementations. Both a FRF and a time signal are highly sampled vectors (around 8k and 12k respectively) and therefore some segmentation must be implemented in these vectors in order for the feature recognition and extraction performed by a CNN to be successful. Moreover, performing this segmentation with segment overlap, an original FRF can be split into a big amount of segments, which will generate more data. Since for the application of DL algorithms, having large datasets is critical to have good accuracies, this pre-processing phase is a step forward to meeting this requirement.

One more is performed in the pre-processing phase of the algorithm, which is shuffling the data. The aim of this operation is to randomly select certain FRFs from the original dataset for the training phase of the 1D CNN, thus removing possible biasing on the overall algorithm procedure, since the defected and non-defected FRFs to train the CNN won't all be from nearby node measurements. The previously described operations can be sequentially presented as:

1. **Shuffling:** Randomly mixing the FRFs in order not to have biased CNN trainings;
2. **Segmentation:** Splitting the FRF into equally sized segments;
3. **Overlap:** Introducing a certain shift for the segmentation procedure to generate more segments and therefore data from an original FRF.

## 1D CNN

Differently to what was done for the 2D CNNs, the use of 1D CNNs both for time and frequency data procedures, involved the crafting of the network's architecture. The same architecture that was used for the 1D CNN in the time domain, was also used for the frequency domain which is summarized in table A.1 of the appendix A. Its architecture can then be summarized as 5 convolution, batch normalization, ReLU, pooling blocks, followed by two fully connected layers with one dropout layer, ending with a softmax and classification layer. Further information can be found in sub-section 7.4.

### 8.3.2 Results

Figure 8.13 shows the 1D CNN results on the FRFs. This defect map portrays the classification of each FRF measured for each of the nodes, in the LDV measured grid. The FRFs which were classified as non-defected are associated to the points in blue, and the ones classified as defected are associated to the points in yellow. Nine groups of yellow points were detected. Seven correspond to the location of the four squared defects, plus three bigger circular defects. One group, in the bottom left corner of the plate matches the location of the PZT patch and therefore the location of the excitation to the plate. The last group contains two defected nodes, in the middle left part of the plate and represents a misclassification, since no defect is there.



**Figure 8.13:** 1D CNN for frequency-domain data result.

Considering the training of the 1D CNN, it was done using the label map represented by figure 7.12. The CNN trains with 50 % of these labels, and performs classification on the entire dataset. This map showed evidence of five defects, therefore the classification of the CNN provided the accurate identification of two more defects, plus the PZT location, showing the algorithm’s ability for feature learning and feature extraction.

### 8.3.3 Bayesian Optimization

Equally to the optimization procedure developed for the 1D CNN for the time domain, the Bayesian Optimization was also applied for an optimization of the 1D CNN applied to analyse the FRFs. This is due to the same problem of having the need to tune many parameters for its training and also to define its architecture. To sum up, the aim is to obtain the best as possible architecture to perform an optimized feature extraction and feature classification and to obtain the best set of training parameters, in order for the neurons of such a CNN to be well tuned to this damage detection classification problem.

In subsection 7.4.3, the conundrum behind this optimization application, the dichotomy between optimizing training hyperparameters of the Stochastic Gradient Descent with Momentum algorithm for the back-propagation and optimizing the CNN architecture to have a good solution of feature extraction and classification, are more extensively explained. For the purpose of not being repetitive, in this subsection, a more brief explanation of the optimization procedure, including the presentation of the parameters left to be optimized, will be given. Thus, there can be summed up that nine parameters were given for the Bayesian Optimization, four regarding the training and five regarding the architecture, which are represented in table 7.1, along with the optimization intervals.

Table 8.1 shows the results for the training parameters. Table A.3 (appendix section A) represents the scheme of the CNN architecture built upon the optimized parameters.

**Table 8.1:** Optimized training hyperparameters for the frequency-domain 1D CNN

Training Hyperparameters	Optimized Values
Learning rate	0.0007
Learning rate drop factor	0.4995
Stochastic gradient descent momentum	0.8744
L2 regularization parameter	7.2038e-9

There can be seen that, except for the dropout factor, the optimized CNN architecture obtained for the frequency-domain is equal to the one obtained for the time-domain. Thus, the same solution for

the CNN layers, for feature extraction with the convolutional layers and for feature classification with the fully connected layers, was found. In comparison, to the hand-made 1D CNN used to present the results on the PMMA plate measurement (architecture in table A.1), the optimized architecture (table A.3) has less convolutional layers and more fully connected layers. Thus, the optimization procedure found that more emphasis should be given to the classification part and less emphasis should be given to the feature extraction part. Moreover, it can be said that it is curious that a very similar configuration of the optimized CNNs was found for both the time and frequency domain. However, although not presented, a few more optimal architecture configurations were found to give the same accurate result as the architecture configuration presented in this subsection.

In order to test the optimized CNN, a dataset with around 3700 measured points was analysed, thus having a bigger resolution of measured points per defect. The labels of such a dataset, given for the Bayesian Optimization as the best result mark, were obtained from manipulating results obtained with the ML tool, and are presented in picture 8.14b. Figure 8.14a shows the classification results.



(a) Classification result from the optimized frequency-domain CNN.

(b) Labels for the Bayesian Optimization.

**Figure 8.14:** Figure a) represents the results obtained with the optimized 1D CNN on the PMMA plate, which was obtained from the Bayesian Optimization in relation to the labels in figure b).

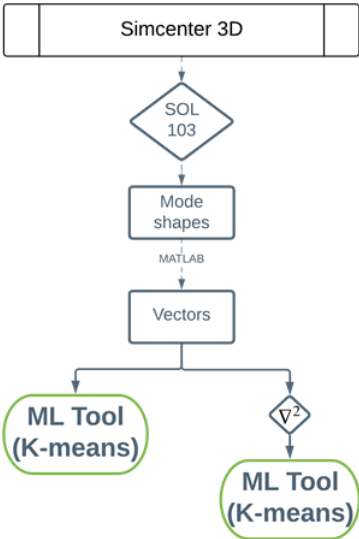
The optimized CNN accurately classified 7 defects, at the same time detecting the piezopatch’s location in the middle of the plate. Having given positive classification results, there can be concluded the successfulness of the optimized architecture for this damage detection purpose, along with the success of finding good hyperparameters for its training using the Bayesian Optimization.

## **Part III**

# **Defect Detection for Simulation Data**

Secondly to developing an algorithm based methodology for damage detection in experimental data, some of the previously applied techniques were also implemented for the same purpose in data obtained through simulation. The aim is to study the robustness of the applied techniques to detect damages with different depths and with different number of information points per defect. The simulations performed in this part of the thesis project were done using the Siemens Simcenter 3D<sup>®</sup> software.

Similarly to part II of this thesis project, the part III algorithm-based methodology uses Machine Learning (ML) techniques for damage detection with a similar procedure as the one used for the mode shape analysis, in the frequency-data procedure. Whereas previously the mode shapes were obtained from the Polymax tool of the Siemens Simcenter TestLab<sup>®</sup> software, in this part, they are simulated with SOL 103 - Real Eigenvalues of the Siemens Simcenter 3D<sup>®</sup> software. Figure 8.15 shows the developed methodology, where the algorithm is presented in the ending of each column as the final procedure step. One Machine Learning (ML) tool was developed to analyse data from mode shapes and their 2nd Gradients. The full description of the simulation steps until the calculation of the mode shapes will be presented in this part, along with the algorithm's methodology and results. In the end of this part, the robustness analysis performed with this algorithm will also be presented.

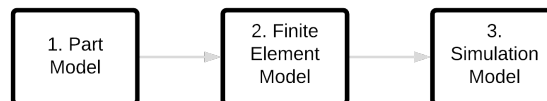


**Figure 8.15:** Algorithm methodology developed in this thesis project for simulation data.

# 9

## Simulation Setup

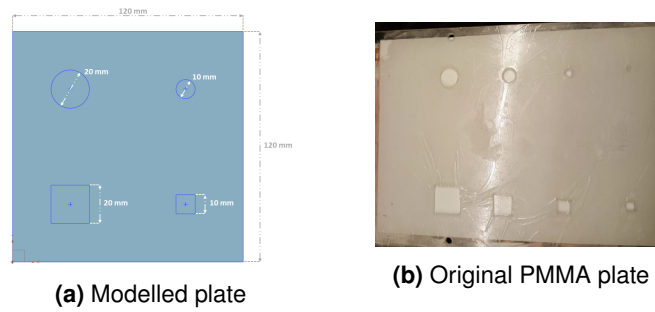
As mentioned before, the simulations were performed using the SOL of the Siemens Simcenter 3D® software. With it, the mode shapes were calculated modes within the chosen frequency interval. Therefore, the input to the simulation will be a plate modulated with defects, to which there were assigned meshes and the output will be the mesh node displacements according to the specific modes. Considering the frequency interval, following the trend of the experimental data, it was chosen to be from 0 to 40 *kHz*, once again according to the Local Defect Resonance (LDR) concept in order to retrieve information regarding high frequency modes that contain resonance patterns dominated by the defect region vibration. In this section, the overall procedure to create the simulations will be explained, from the creation of the plate part model, to the finite element model and ending with the simulation results (Figure 9.1).



**Figure 9.1:** Overall simulation procedure.

### 9.1 Part Model

One of the most tested plates in this project was the PMMA with 8 damages (4 circular and 4 squared). Therefore, following the trend of simulating something related to what was tested experimentally, the modelled plate is a simpler version of the original PMMA plate which contains two squared and two circular FBHs. In figure 9.2a a comparison between the modelled and tested plate is presented. The FBH dimensions in the modelled plate were made so to resemble the original ones. Moreover, table 9.1 presents a sum-up of the modelled plate dimensions.



**Figure 9.2:** Match between the modelled and the real plates. Both have similar Flat Bottom Holes sizes and shapes

**Table 9.1:** 2D part model properties

2D part model properties	
squared plate	120x120 mm
2 squared FBH	20 mm & 10 mm
2 circular FBH	20 mm & 10 mm

## 9.2 Finite Element Model

When passing from the part to the finite element model, one more consideration regarding the comparison with the experimental procedure was included. Being that the grid of points measured by the LDV contains equidistant points in a regular grid (figure 6.1b), the finite element applied in the meshing process was the *CQUAD4*, which contains 4 degrees of freedom (DOFs) and better resembles a regular grid, than an element with a non-rectangular shape or with more DOFs (figure 9.3a). Moreover, the material used for the simulation model was equally PMMA.

In the process of attributing meshes to the part, meshes with equal sizes were attributed to all the surfaces of the plate, this is the FBH and the plate. A link between all meshes was established in order for the border DOFs of the plate mesh to match the meshes of the FBHs. This way a meaningful finite element model was designed, that portrays a trustworthy simulation to obtain correct mode shapes.

With the purpose of making a robustness analysis of the Machine Learning (ML) algorithm, many thickness were attributed to the FBH meshes along with variant mesh sizes. The table 9.2 presents the different mesh sizes and thickness variations used in this robustness analysis.

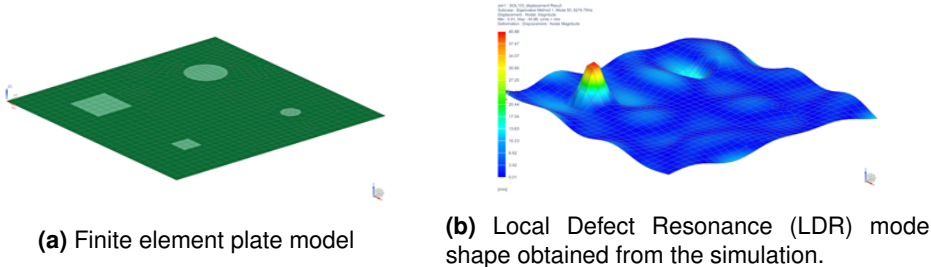
**Table 9.2:** 2D finite element model properties

2D finite element model properties	
element	<i>CQUAD4</i> {8, 4, 2, 1} mm
material	PMMA (plate thickness 5 mm)
FBH	thickness {10, 30, 50, 70, 80, 90} %

## 9.3 Simulation Model

Finally, having a correctly constructed part and finite element model, the simulation analysis was performed after. Using SOL 103 - Real Eigenvalues, the mode shape displacement information was ob-

tained for a similar frequency band as the one used for the PZT excitations, which is  $[0, 40000]Hz$ . Depending on the mesh size and the FBHs thickness a different amount of mode shapes was obtained in each simulation. For the highest mesh size (8 mm) around 600 mode shapes were obtained in this frequency band, but diminishing the mesh size, this number converged to 400. The finite element model in was simulated in free-free boundary conditions and figure 9.3b shows one obtained mode shape with Local Defect Resonance (LDR) behaviour.

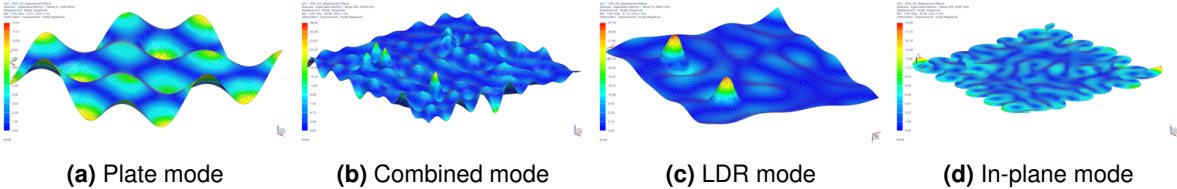


**Figure 9.3:** Finite Element Analysis.

**Table 9.3:** Simulation model properties

SOL 103	
constraints	<i>Free-free conditions</i>
Frequency band	$[0, 40000]$ Hz

Many mode types were obtained in such a big frequency band. The smallest frequencies are dominated by plate modes, but as the modal frequencies increase, so do the number of LDR, in-plane and combined modes. Being that the LDV in the experimental procedure measures the out-of-plane displacement, no in-plane modes can be calculated for the experimental procedure and therefore, these modes were excluded from the analysis presented in this chapter. A priori, one can expect that the bigger the thickness of the FBHs, the bigger amount of plate modes and less amount of LDR modes there will be in the simulation. This will present a challenge for the algorithms detectability capacity, as will be presented in the next chapter according to the robustness analysis.



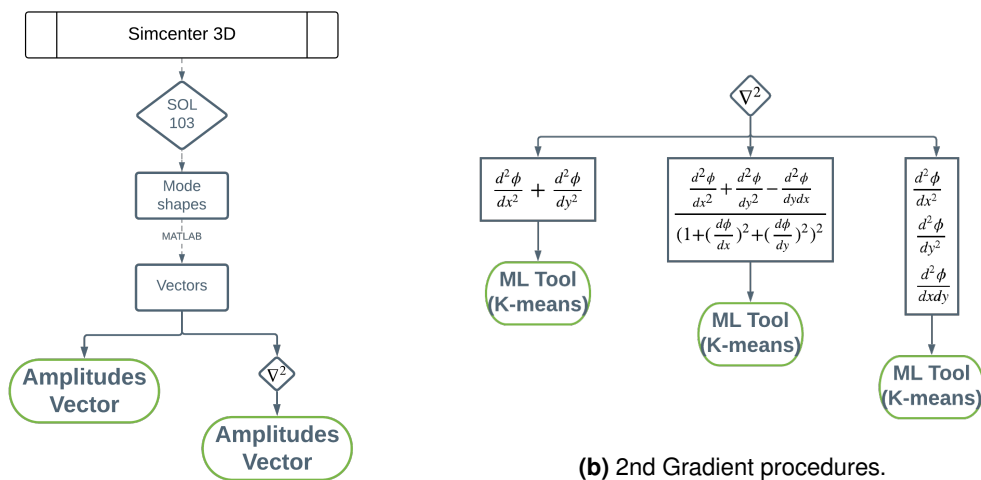
**Figure 9.4:** Examples of obtained modes.



# SOL 103 - Real Eigenvalues Procedure

This chapter contains the description of the algorithm related to the SOL 103 data and also a brief description of the data itself. The algorithm developed uses the Machine Learning (ML) technique K-means, just as shown in image 8.15.

Moreover, as can be seen by the overall methodology in figure 8.15, after the mode shapes being calculated from the SOL 103 - Real Eigenvalues, one other operation is performed previously to one of the Machine Learning Tool implementations. This operation consists of applying 2nd Gradient techniques to the mode shape data in order to highlight the defect's influence. Three different procedures were tested which are shown in figure 10.1b and can be distinguished in calculating the Laplacian(left procedure), the Gaussian Curvature (middle procedure), and the second Derivatives (right procedure). In the end, three different datasets are created and input to the Machine Learning Tool. As a note, this process was done equally to the one implemented in the 2nd Gradient branch of the frequency data procedure in the experimental part.



(a) Simulation data used for each algorithm.

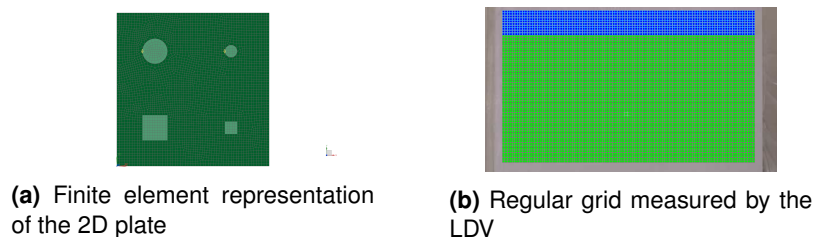
Figure 10.1: Overview of the simulation data and methodologies used for the 2nd Gradient techniques.

## 10.1 Simulation data

The Machine Learning tool developed for the simulation-domain data analyses two types of input, the mode shapes and their 2nd Gradients. Both are represented as amplitude vectors, as can be seen in figure 10.1a and were calculated for an interval of 0 to 40  $kHz$ . The choice of such a big interval of frequencies is compliant with the Local Defect Resonance (LDR) concept, in order to obtain mode shapes dominated by the defect vibration behaviour, which appear for high frequency modes. The application of the 2nd Gradients was done in MATLAB®.

### 10.1.1 Mode shapes

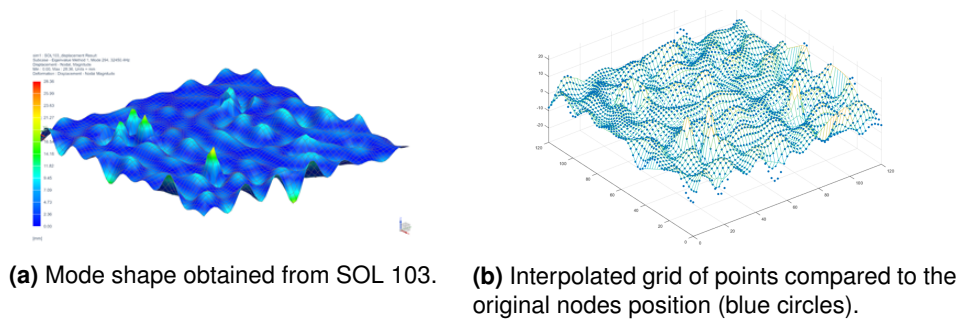
One of the challenges of applying algorithms similar to the ones developed in the experimental procedure to simulation data is the difference between data measured with a regular grid (such as the one by the LDV) and finite element simulations with finite elements that are not regularly distributed with perfect rectangular shapes. In terms of applying 2nd Gradients in a two-dimensional grid of points, this operation is performed with much increasing difficulty for non-regular grids of points. Therefore, although for the mode shape data, no gradients were applied, since for the 2nd Gradient algorithm an interpolation process was needed to generate a regular grid, for this data also an interpolation step was performed.



**Figure 10.2:** Difference between a regular grid of equidistant points as the one performed by the LDV, and a grid of points obtained by the finite elements' DOFs.

The interpolation process was done with a *cubic* interpolation to obtain a grid of point with a 1:1 reasoning to the finite element model. This is, in case the finite element model had perfect squared elements, the grid of points would perfectly correspond to the one obtained from the interpolation, thus no increase of points was done by the interpolation process in order to maintain the meaning of the mode shape using that specific mesh size. Of course that it would be impossible to obtain a regular equidistant grid of points from the finite element model, since there are circular shapes, other squared shapes inside the original plate. Therefore, considering a certain mode shape as the one represented in the figure 10.3a, the interpolated result is represented in figure 10.3b by the grid of lines (the points in the interpolation image correspond to initial position correspondent to the finite element model calculation).

As can be seen by the resemblance of the interpolated grid and original shapes, this process conserves the meaning of the mode shape, at the same time not adding any additional information as to the mesh size used in the original simulation. The amplitudes from the regular grid will be arranged in a vector of points and will be the input for the Machine Learning Tool.

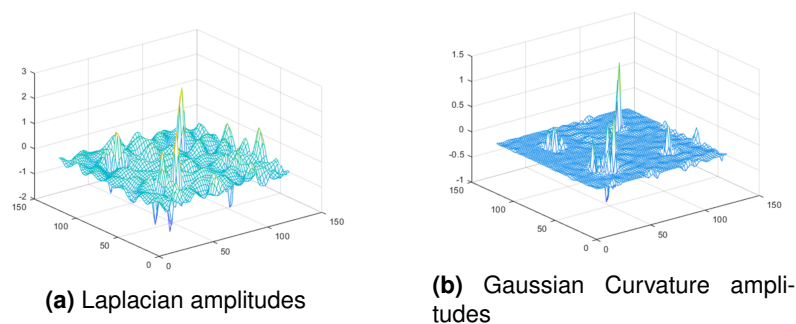


**Figure 10.3:** Comparison between the irregular simulation and the regular interpolation grids.

### 10.1.2 2nd Gradients of mode shapes

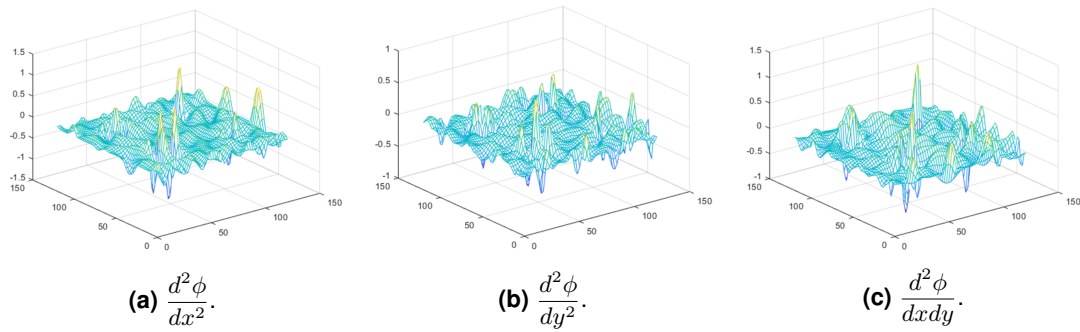
Aiming to highlight the defect's presence in a mode shape and following the same procedure as the one performed for the frequency data procedure, 2nd Gradient techniques were implemented. Of these, there can be distinguished: the Laplacian, the Gaussian Curvature and the 2nd Derivatives. For each of the 2nd Gradient techniques summed in figure 10.1b, there is a correspondent dataset made of vectors of amplitudes, which will be input into the Machine Learning tool.

Figure 10.4 shows the effects of applying the Laplacian (figure 10.4a) and the Gaussian Curvature (figure 10.4b) techniques in reference to a mode shape represented in figure 10.3b. The Laplacian achieved some, but not a significant success in highlighting the presence of defects, comparing to the results obtained with the Gaussian Curvature. The latter demonstrates again to be a strong tool for highlighting the defects' presence in a mode shape with highly combined defect and plate motion. However, as will be seen later, it doesn't mean that it provides the better dataset for damage detection with the Machine Learning tool.



**Figure 10.4:** Plot of the Laplacian and Gaussian Curvature amplitudes with reference to the mode shape interpolation into a regular grid, presented in figure 10.3b.

Figure 10.5 shows how the individual 2nd Derivatives can slightly highlight the defects' presence, but not better than the two previous versions. What highly distinguishes the dataset obtained with the 2nd Derivative in comparison to the previously presented two techniques, is the fact that each of the 2nd Derivatives is analysed separately, whereas for the previous cases, only the final calculation result is submitted in the algorithm.



**Figure 10.5:** Plot of the 2nd Derivatives amplitudes with reference to the mode shape ( $\phi$ ) interpolation into a regular grid, presented in figure 10.3b.

## 10.2 Machine Learning Tool (K-means)

The algorithm implemented to analyse the data obtained with simulation implements the K-means clustering technique presented in the theoretical Chapter 3 Machine Learning. Comparing to the Machine Learning tool developed for the frequency data procedure in section 8.2, it implements only the clustering technique out of the two ML techniques implemented by the ML tool (GADAC). This is due to the fact that from SOL 103 - Real Eigenvalues, the mode shape displacement amplitudes are real values, whereas, from Siemens Simcenter TestLab<sup>®</sup> calculations, they are complex values, and the Gaussian Anomaly Detection Automated by Clustering (GADAC) technique needs both real and imaginary input. Nonetheless, the algorithm that will be presented in this section provides good and meaningful results, since the K-means algorithm by itself is the first step into building a defect map in the GADAC algorithm as well. In the end, this algorithm provides a fast tool to analyse hundreds of mode shapes and determining whether the plate contains damages or not.

### 10.2.1 Pre-processing

As mentioned previously in sub-section 10.1.1, a pre-processing step of interpolating the mode shapes into a regular grid was needed for the implementation of the Machine Learning tool. Along with the *cubic* interpolation calculation one more operation has to be performed in the data pre-processing step, which is removing the in-plane modes. The LDV experimental measurements register only the out-of-plane motion, hence no information can be inferred for the experimentally calculated mode shapes regarding in-plane vibrations. To maintain coherence with the experimental procedure developed in this project, the simulation data analysis did not include the analysis of the in-plane mode shapes calculated from the finite element model. Figure 10.6 represents the context and steps of the pre-processing within the overall procedure used for the analysis of the simulation data obtained with SOL 103, until the calculation of the total defect maps which contain the defect detectability results. As portrayed, after the removal of in-plane modes and interpolation calculation, the interpolated mode shapes and their 2nd Gradients are analysed by the Machine Learning tool.

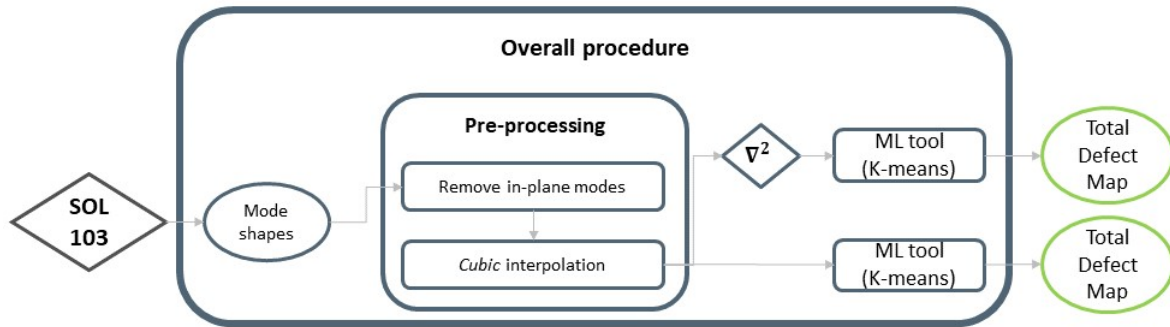
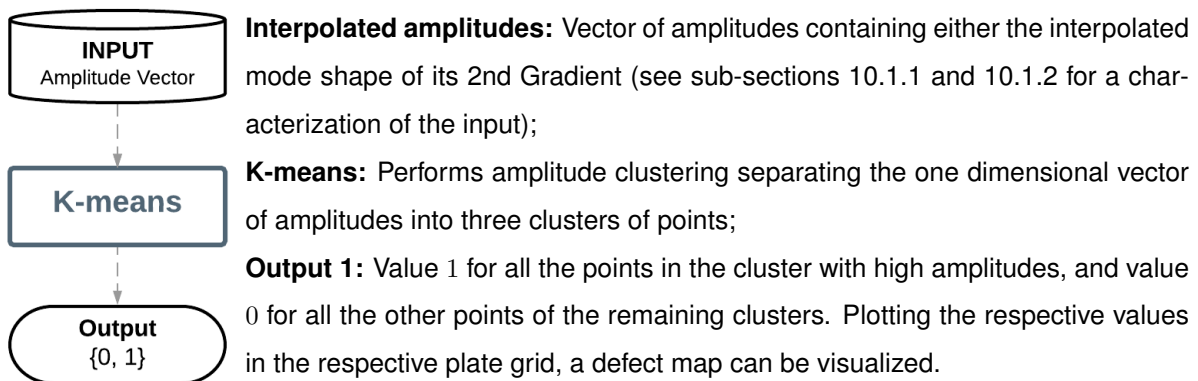


Figure 10.6: Pre-processing context inside both Machine Learning tools procedure.

## 10.2.2 Methodology

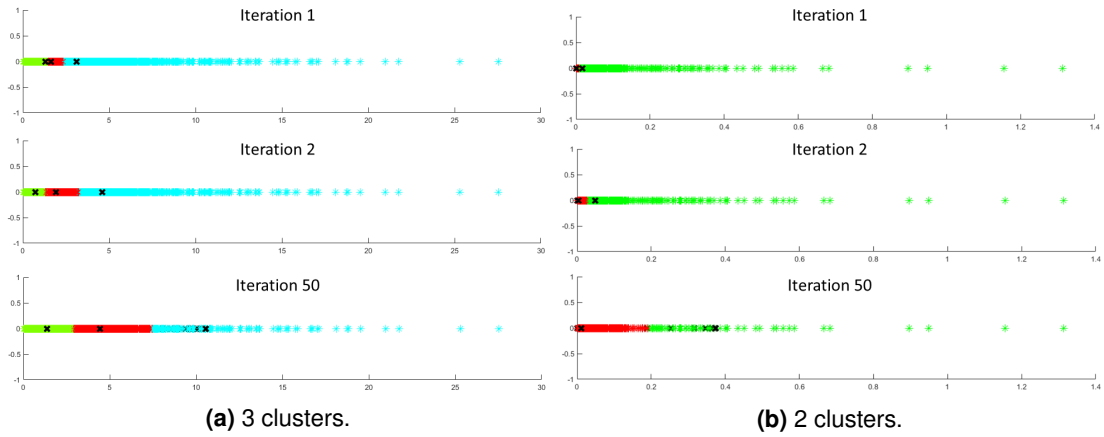


### K-means

K-means was chosen to run with 3 clusters and with 50 iterations which were experimentally found to guaranty this algorithm's convergence. Being the iteration process finished, the points of the cluster with highest amplitudes are assigned to the defected nodes in the plate, according to the Local Defect Resonance (LDR) principle. Three clusters were chosen instead of two, because since there are many combined modes, three cluster do a better job to separate the high amplitudes of defect vibration than two clusters. For the analysis of the Laplacian and 2nd Derivatives versions of the second gradient, the number of clusters was also chosen to be 3, but for the Gaussian Curvatures it was chosen to be 2. Since this latter is a powerful tool to highlight the presence of a defect in a mode shape, the application with 2 clusters was found to be sufficient. Figure 10.7 shows the clustering process with both two and three cluster and the respective results after the fifty iterations.

## 10.2.3 Results for the mode shapes

After all steps of the Machine Learning tool (K-means) applied to the mode shape data, the result will be a defect map, from which there can be or not visualized defect presences marked by being detected multiple times across the various mode shapes from the full mode shape calculation from the 0 to 40kHz frequency band. Thus, for each mode shape the algorithm outputs a defect map, which will be summed to the previous detected maps of previous mode shapes already ran through the algorithm in

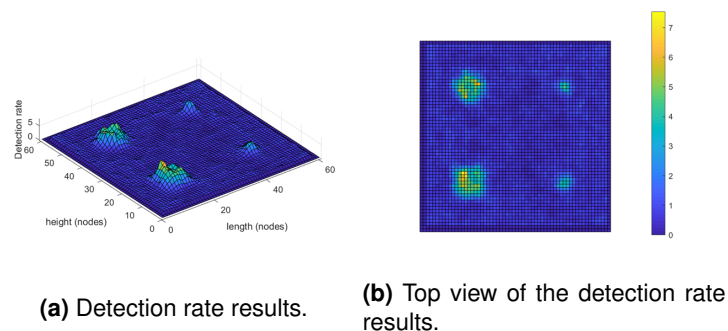


**Figure 10.7:** Visualization of the clustering process

an automatic process. Information regarding the independent classification of one mode shape can be found in the appendix section B.2.

The classification result of the Machine Learning tool on all the mode shapes calculated for a plate with a defect thickness of 30 % (plate has 30 % of thickness in the defect locations) and using a mesh size of 2 mm is presented in figure 10.8. These results are presented in terms of detection rate (equation 10.1). The detection rate is a calculated metric which correlates the detection for the damaged plate in comparison to the detection of the algorithm for an healthy plate (plate has 100 % of thickness in the defect locations). Therefore, the detection rate means that a particular node was detected by the Machine Learning tool that number of times more for the defected plate than for the healthy plate. As can be seen, the detection of all 4 defects was successful. Moreover, information regarding their shape could also be retrieved.

$$Detection\ rate = \frac{total\ defect\ map\ from\ plate\ with\ FBH\ thickness\ X\%}{total\ defect\ map\ from\ healthy\ plate} \quad (10.1)$$

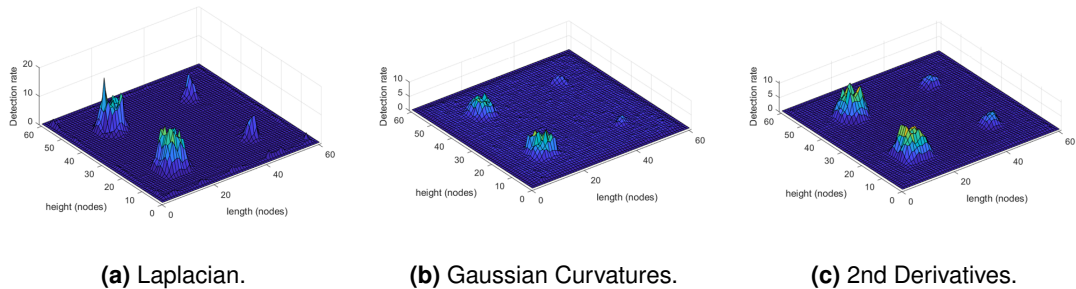


**Figure 10.8:** Results for a modelled plate with 30% thickness and a mesh size of 2 mm.

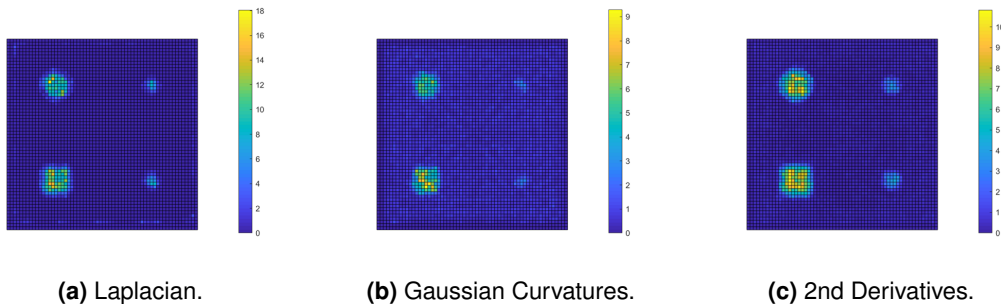
### 10.2.4 Results for the 2nd Gradients of the mode shapes

For each interpolated mode shape obtained after the pre-processing step, three 2nd Gradient techniques were applied: the Laplacian, the Gaussian Curvature and 2nd Derivatives, as explained in sub-section

10.1.2. The Machine Learning tool analyses each version of the 2nd Gradients and outputs a total defect map which is divided by the defect map obtained for the mode shapes of a healthy plate, in order to calculate the detection rate (equation 10.1). In this sub-section the results for a plate modelled with 30 % of thickness and analysed with a mesh size of 2 mm (equally to the dataset used to present the results in figure 10.8) are presented for each version of the 2nd Gradients in figures 10.9 and 10.10. Moreover, information regarding the independent classification of one 2nd Gradient of a mode shape can be found in the appendix section B.2.



**Figure 10.9:** Detection rate results of the Machine Learning tool applied to the 2nd Gradients of the mode shapes calculated for a modelled plate with 30% thickness and a mesh size of 2 mm.



**Figure 10.10:** Top view of the detection rate results of the Machine Learning tool applied to the 2nd Gradients of the mode shapes calculated for a modelled plate with 30% thickness and a mesh size of 2 mm.

All the defects were accurately detected by the algorithm for all versions. In a comparative evaluation, the 2nd Derivatives results are slightly better in terms of capturing not only the defects with a good resolution for the non-damaged locations in the plate, but also to capture their shapes. A deeper study to determine the best 2nd Gradient version will be presented in the next sub-section.

### 10.2.5 Best version of the 2nd Gradient algorithms

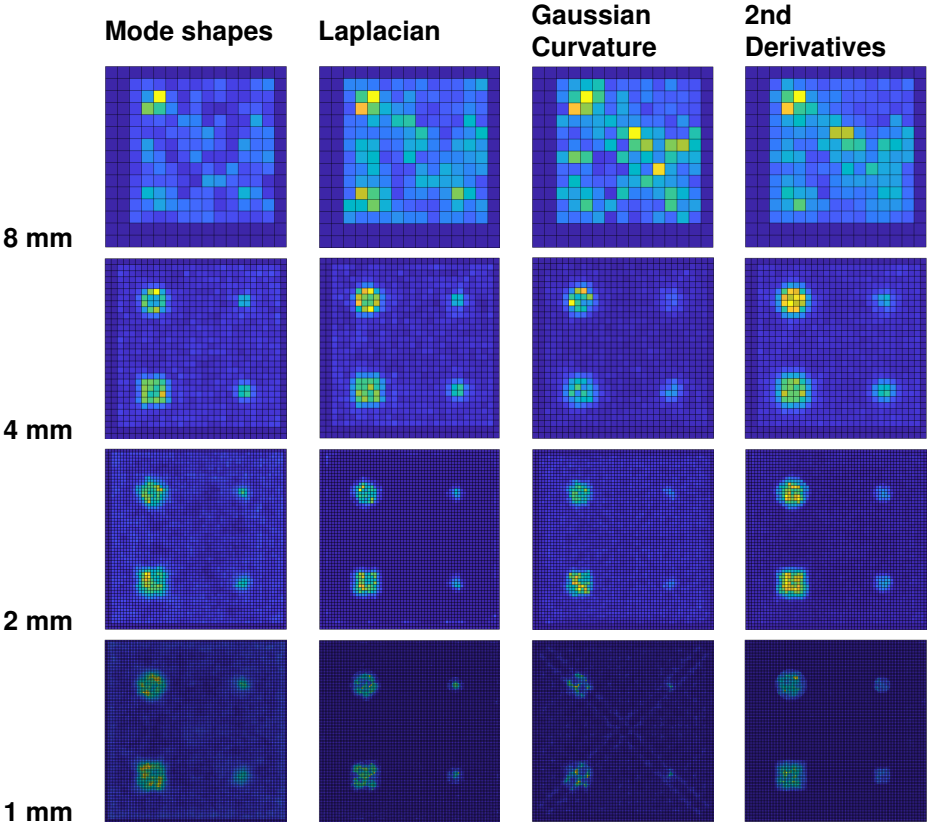
At this stage, three different versions of 2nd Gradient techniques have been presented. Between the gradient calculations and implementations, all method have considerable differences, and thus a study can be performed as to identify the best version. In this subsection that study will be presented, by evaluating the detection rate results (equation 10.1) of the Machine Learning tool algorithm. By calculating the detection rate a visualization considering the number of times a certain node was detected as

defected in comparison to the number of times it was detected as defected from the healthy case (Flat Bottom Hole (FBH) = 100 %) can be obtained. The premise is that a node correspondent to some defect will be detected a big amount of times more than for the healthy case.

Table 10.1 presents the results for many mesh sizes (as described in table 9.1: 8, 4, 2, 1 mm) applied to a modelled plate with 30 % Flat Bottom Hole (FBH) thickness, for each dataset version (mode shapes; Laplacian; Gaussian Curvature; 2nd Derivatives). The Machine Learning tool applied to the mode shapes is presented as a reference case. The objective will be making a visual analysis of the results obtained for each algorithm and possibly make some conclusions considering the best version of the Machine Learning tool applied to the 2nd Gradient of the mode shapes, in order to select that one for the following robustness analysis, which will be presented in the next chapter.

Regarding each of the images in table 10.1, the boundaries of the plate were removed from the analysis (dark blue columns and rows at the edges of each image), due to the error induced by the application of the 2nd Gradient operation to the limits of the plate (for coherence this was also applied to the results of the ML tool - mode shapes).

**Table 10.1:** Results for the modelled plate with FBH thickness = 30 %



A first conclusion taken from observing the results in table 10.1 is that the defect identification was successful for all mesh sizes except the 8 mm. Secondly, taking a closer look into the images from ML tool -1/2/3 that correspond to the 2nd Gradient versions, there can be seen that the 2nd Derivatives version provides the best defect maps for all mesh sizes. This is, it is the one which obtains a better resolution considering the defect identification and more accurately provides information on the defect



shape. Therefore, the ML tool - 3 will be considered out of the three 2nd Gradient ML tool versions, for the robustness analysis that will be presented in the next chapter.

### 10.3 Robustness analysis

According to table 9.2, in total 24 plate conditions were modulated and used for the robustness analysis. More precisely, plates were modulated with 6 different defect thickness and for each of those models, 4 mesh sizes were used. The aim of varying the mesh size and defect thickness is to study the boundaries of detection capability with the ML tool. Figure 10.11 presents the robustness analysis methodology, considering the analysis of different plate conditions by both the ML tool version to analyse mode shapes and its 2nd Derivatives. This test includes a total of 48 runs, since for each of the 6 thickness, 4 mesh sizes were applied and the results ran through 2 algorithms.

The FBH thickness was varied by manipulating the plate thickness in the defect regions. The full plate has 5 mm of thickness, therefore a defect with 50 % thickness means the plate has 2.5 mm in the corresponding sections. Considering the different mesh sizes, for coherence principles, the same mesh size was given to all surfaces of the plate on the damages and non-damaged regions.

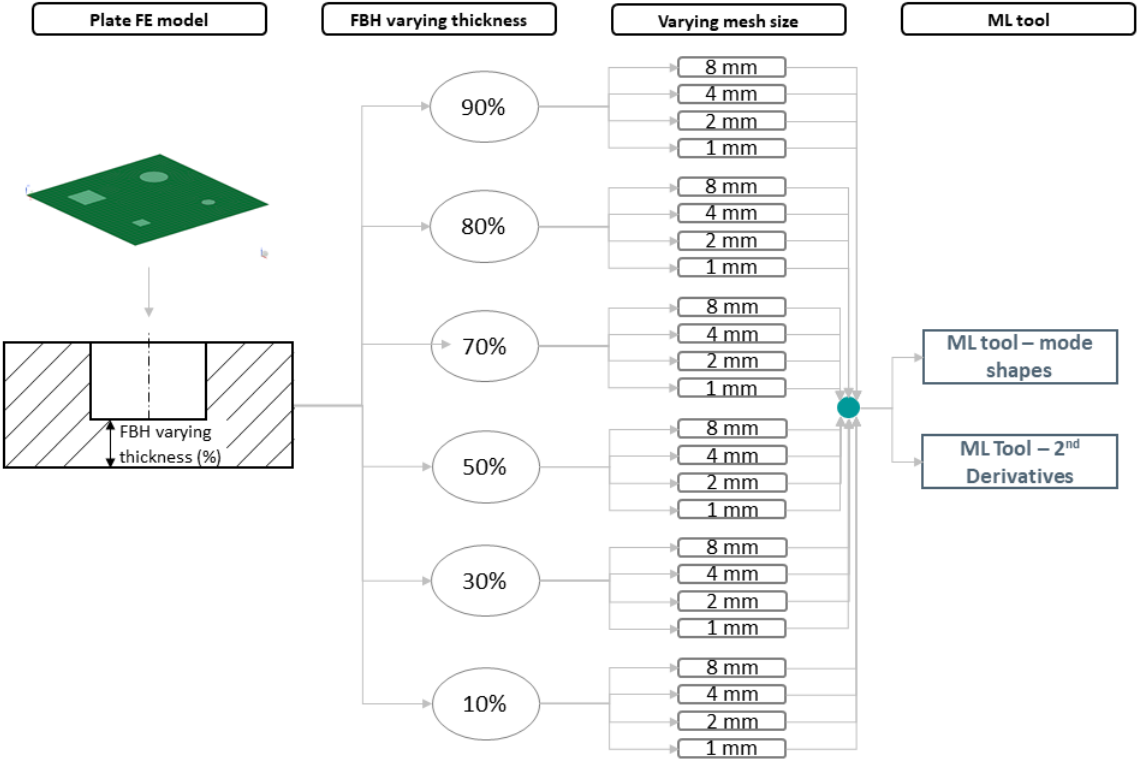


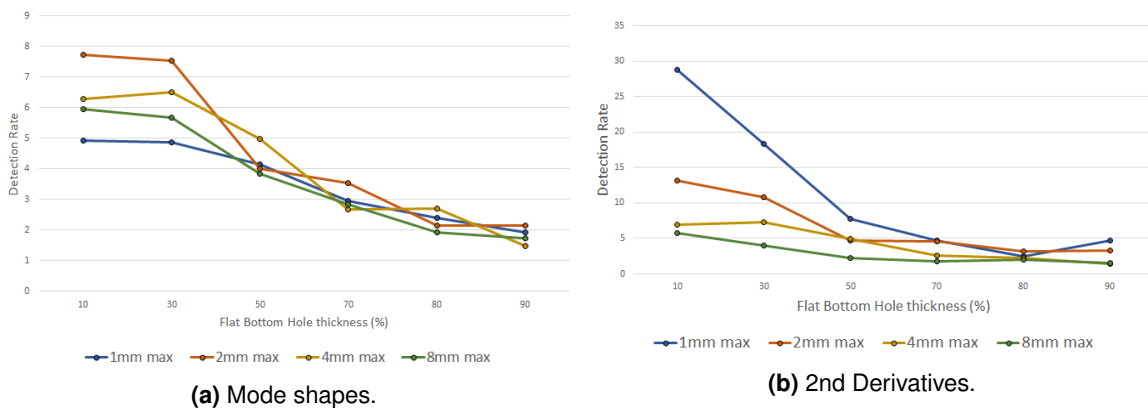
Figure 10.11: Robustness analysis methodology.

Table 10.2 contains the results of the robustness analysis, presented in terms of detection rate (equation 10.1). It is organized as follows: each row corresponds to a certain FBH thickness plate configuration; each column to a corresponding mesh size; and for each column and row entry, two images are displayed which are the ML tool detection rate results for the mode shapes version (on top) and the 2nd

Derivatives version (below). Many important conclusions can be taken, from analysing the results in this table, regarding limitations of the ML tool's capability to detect Flat Bottom Holes (FBHs).

There is a better damage detection resolution as there is a decrease in the defected region thickness and mesh size (moving downwards and rightwards in the table). This means that the more prominent a defect is and the more points there are in the mesh per defect, the better is the detection by the algorithm. Other conclusions for the results presented in table 10.2 are:

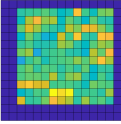
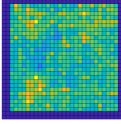
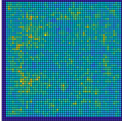
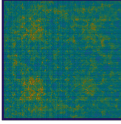
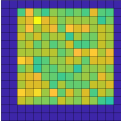
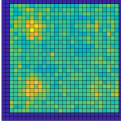
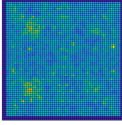
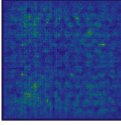
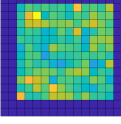
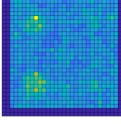
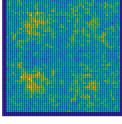
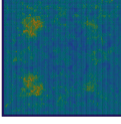
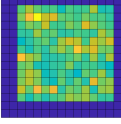
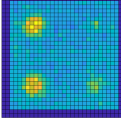
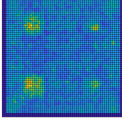
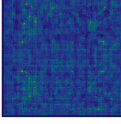
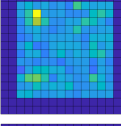
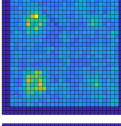
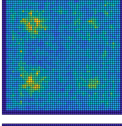
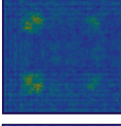
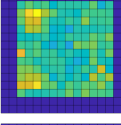
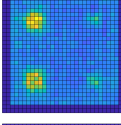
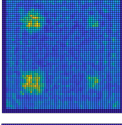
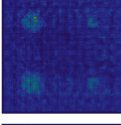
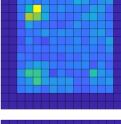
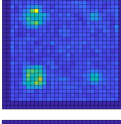
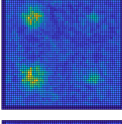
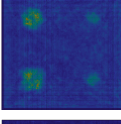
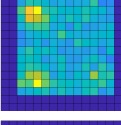
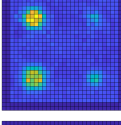
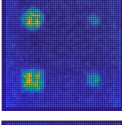
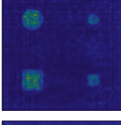
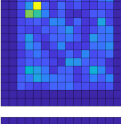
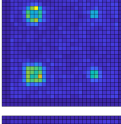
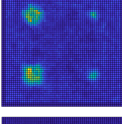
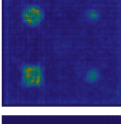
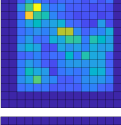
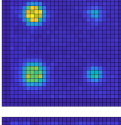
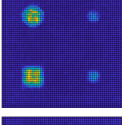
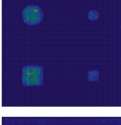
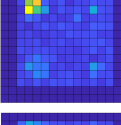
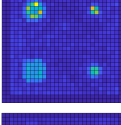
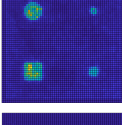
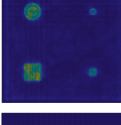
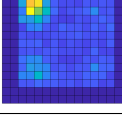
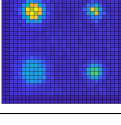
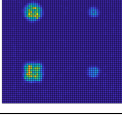
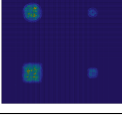
- Regarding the ML tool's detection capability for the variant FBH defect thickness shown in the evolution of results through the rows, a gradual increase of defect's resolution is achieved as the FBH thickness decreases (meaning more loss of plate thickness in the defects' areas). From the images for the plate configuration with a FBH thickness of 90 %, the damages can be hardly distinguished in the plate. For the configuration with 80 % FBH thickness, a better defect resolution can be observed, especially for the 2nd Derivatives prediction regarding the 4 and 2 mm mesh sizes. Hence, there was concluded that the threshold for defect detection performed by this algorithm is 80 %.
- Regarding the detection capability for variant mesh sizes shown in the evolution of results through the columns, a general improvement of defect resolution happens from the 8mm to the 4mm and to the 2 mm mesh sizes. However, no significant improvement can be observed considering passing from a 2mm to a 1mm mesh size. Moreover, a note can be taken regarding the results with the 8mm mesh not having a meaningful damage detection. This is associated to the fact that this mesh size is similar to the size of the smallest defects (10 mm), therefore there are not enough points per defect in order to detect them.
- Regarding a comparison of both versions of the ML tool, the mode shapes and the 2nd Derivatives, a conclusion can be taken for the better performance of the 2nd Derivatives version. For plates with higher FBH thickness, the results with this version capture better the damage locations and shapes in comparison to the results obtained with the mode shapes version.



**Figure 10.12:** Maximum detection rate values for the ML tool applied to a) mode shapes and b) 2nd Derivatives of the mode shapes.

Important conclusions listed above were taken regarding the ML tool's robustness to detect damages

**Table 10.2:** Robustness analysis results

mode shapes 2nd Derivatives FBH thickness	mesh size			
	8mm	4mm	2mm	1mm
90 %				
				
80 %				
				
70 %				
				
50 %				
				
30 %				
				
10 %				
				

from the analysis of the detection rate result images in table 10.2. Additionally, the maximum value of detection rate was registered for each of the ML tool's results for each plate configuration. These values are shown in figures 10.12a and 10.12b, respectively for the mode shapes and 2nd Derivatives versions. Both figures show a graph with the evolution of these values for the varying FBH thickness (x-axis) and registered with the different mesh sizes (each curve corresponds to the results obtained with one of the four mesh sizes).

The detection rate value means that a particular node was detected by the ML tool that number of times more for the defected plate than for the healthy plate. Overall, the detection rate results for the 2nd Derivatives version in figure 10.12b are higher than the ones for the mode shapes version in figure 10.12a. This goes in agreement with the better results obtained with the 2nd Derivatives ML tool version. In other comparative term, results for both versions show a descending progression of detection rate as the FBH thickness increases, for all mesh sizes. This means that the detectability is decreasing as the thickness in the damaged areas gets closer to the plate thickness.

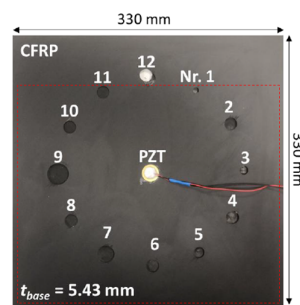
# 11

## Other results

Although across this thesis project document, the same dataset measured by the LDV on the PMMA plate with 8 defects was used to present the results of each algorithm, more plates or other datasets of the PMMA plate with a varying number of defects were measured and evaluated by these algorithms. A Carbon Fiber Reinforced Polymer (CFRP) plate with circularly distributed Flat Bottom Holes (FBHs), a smaller subset of the PMMA plate and another Carbon Fiber Reinforced Polymer (CFRP) plate with equidistantly distributed Flat Bottom Holes (FBHs) were also evaluated by the damage detection algorithms, whose results will be further presented.

### 11.1 CFRP plate 1

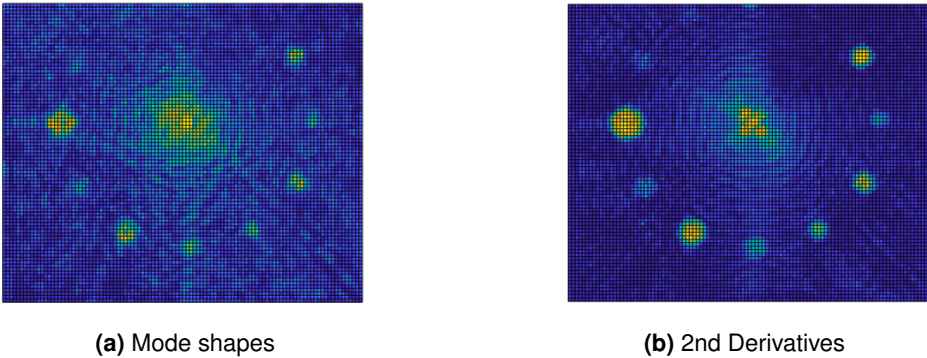
A Carbon Fiber Reinforced Polymer (CFRP) plate which contained 12 circularly distributed Flat Bottom Holes (FBHs) with varying sizes and thickness was measured in this thesis project for an algorithm implementation as well. Since one of the FBHs went through the plate, a smaller set of the plate with the other 11 damages was measured, in order to maintain the coherence of the top surface's vibration measurement. Picture 11.1 shows the distribution of defects along the plate and the LDV measured grid, as the red dashed rectangle. As can be seen, the excitation to this plate was provided by the PZT patch located in the center of the plate, equidistant to the damages.



**Figure 11.1:** 1st CFRP plate's distribution of FBHs and a representation of the measured grid by the LDV (dashed red rectangle).

From the results of the ML tool (GADAC) (figure 11.2), almost all damages were detected from both

the mode shapes and the 2nd Derivatives versions. However, the 2nd Derivatives defect map displays less impact from the detection of the piezoelectric patch's location and consequent excitation, in the center of the plate. This is due to the initial premise that the 2nd Gradient techniques highlight the presence of damages in a mode shape. Moreover, the analysis of these results allowed to define a region in the center of the plate affected by the excitation, which was removed from the analysis of the 2D and 1D CNNs.



**Figure 11.2:** CFRP plate 1 - ML tool (GADAC) results

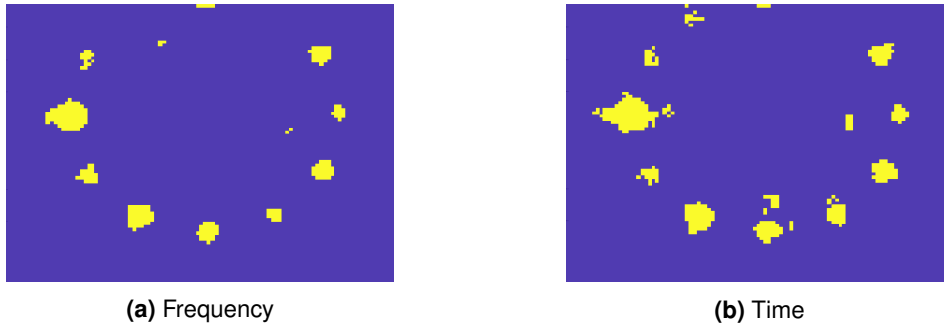
Picture 11.3 shows the results of the 2D CNN applied on the CWT images and of the 2D CNN applied on the STFT images. The former shows less false positives, but also less true positives that the latter, achieving the overall detection of nine defects.



**Figure 11.3:** CFRP plate 1 - 2D CNN results

Picture 11.4 shows the results for both the time-domain and frequency-domain 1D CNNs. The accurate detection of nine defects is achieved for the analysis of the FRFs and ten defects for the analysis of the time-signals. Although the time-domain 1D CNN detected one more defect, a note can be made regarding having also made more misclassifications than the frequency-domain 1D CNN.

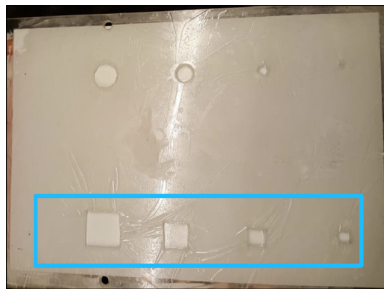
Comparing to damage detection results obtained using weighted vibrational energy calculations for this CFRP plate [39], the results obtained with these algorithms also managed to detect the same number of the plate's defects. Moreover, the implementation of the CNNs allows to obtain binary classification maps, which, although prone to some misclassifications, represent the results with a direct contrast between the non-damaged and damaged locations in the plate.



**Figure 11.4:** CFRP plate 1 - 1D CNN results

## 11.2 PMMA plate

Also considering the PMMA plate with 8 defects, whose results on a singular dataset were presented across this thesis project document, for each algorithm, a measurement on 4 out of 8 defects was performed. This measurement was focused on the squared defects of varying dimension. Picture 11.5 shows the defect' distribution across the plate and a representation of the measured grid of points by the blue rectangle.



**Figure 11.5:** PMMA plate's distribution of FBHs and a representation of the measured grid by the LDV (blue rectangle).

Picture 11.6 shows the results of the Machine Learning tool (GADAC ) applied to the mode shapes and of the Machine Learning tool (GADAC ) applied to the mode shapes 2nd gradient. The results of the former are superior than the ones of the latter. The identification of the four defects, with their shapes and locations was attained.



**Figure 11.6:** PMMA plate (4 defects) - ML tool (GADAC) results

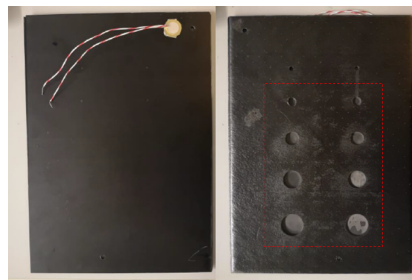
Picture 11.7 shows the results of the 1D CNN applied to the time signals. Since both 1D CNNs have very similar results in terms of detection, only one image was shown. The four defects were detected along with their shapes and locations.



**Figure 11.7:** PMMA plate (4 defects) - 1D CNN results

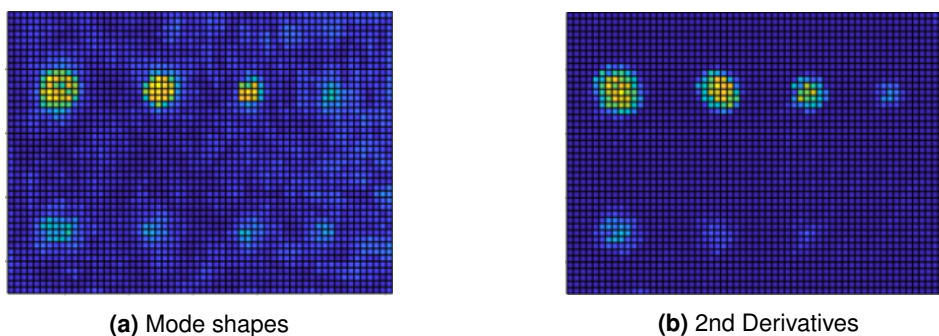
### 11.3 CFRP plate 2

The last plate whose results are going to be presented is a Carbon Fiber Reinforced Polymer (CFRP) plate with 10 Flat Bottom Holes (FBHs) distributed in a grid, two columns of five defects. Each column contains defects with the same thickness, but different dimensions. Of these ten defects, eight were measured by the LDV in such a grid as displayed in picture 11.8. The location of the PZT patch is also shown in the image, and there can be stated that it is outside the LDV measured grid area.



**Figure 11.8:** 2nd CFRP plate's distribution of FBHs and a representation of the measured grid by the LDV (red dashed rectangle).

Picture 11.9 shows the results of the Machine Learning tool (GADAC) applied to the mode shapes and of the Machine Learning tool applied to the mode shapes 2nd gradient. The results of the latter are clearer, but from the former, some considerations regarding the location of the smallest and less thick defect can be taken. Thus there was attained the identification of eight defects for the former algorithm and seven for the latter.



**Figure 11.9:** CFRP plate 2 - ML tool (GADAC) results

Regarding both the analysis of the PMMA and CFRP plate 2, comparing to results obtained with a threshold algorithm on time-domain and frequency-domain data [40], the results shown in this article manage to detect a higher number of these plates' defects. Moreover, the use of ML techniques allowed to build these algorithms with automatic frameworks which can be easily implemented for the analysis of additional datasets.



# Conclusion and Future Developments

After all the research and work developed throughout this thesis, several conclusions were taken. To start, a first statement can be written regarding the success of the algorithm implementation to the purpose of damage detection and the accomplishment of automatic algorithms through the implementation of Machine Learning (ML) and Deep Learning (DL) techniques.

The experimental procedure for damage detection in lightweight plates by measuring the top undamaged surface, was markedly made easier by obtaining thick grids of points, measured in reasonable time. This implementation was made possible using the Laser Doppler Vibrometer (LDV). It surpasses the mass and wire loading of more traditional methods like accelerometers, accomplishing measurements on grids of points with little spacing between themselves, which allow the algorithms to better detect the damages. However, a LDV measures one point at a time, and for grids containing thousands of points, the plate needs to be excited with the piezoelectric patch that same amount of times, which makes for a laborious, although automatic experiment.

Regarding the experimental data analysis part of this project, five different algorithms were created using Machine Learning (ML) and Deep Learning (DL) techniques. Three for the time-domain data and two for the frequency-domain data. Implementing these techniques made possible to create automatic algorithms, to which data is given to be analysed and an output is generated without intermediate manual data interaction. The success of the output is nonetheless dependent on the tuning of the used techniques. The used DL algorithms, Convolutional Neural Networks (CNNs), have a greater need for parameter tuning than the used ML algorithms, which needed only the tuning of two parameters (number of clusters and iterations). Contrary to the 2D CNNs used, the developed 1D CNNs needed both the tuning of training and architecture hyperparameters. In this context, the Bayesian Optimization studied in this project proved that optimal architectures can be obtained with this procedure, which are capable of making accurate classifications.

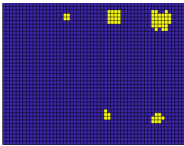
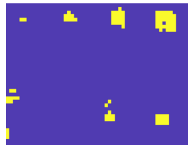


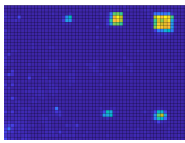
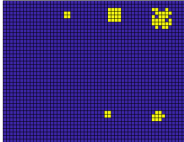
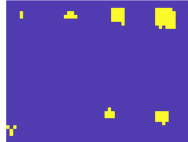
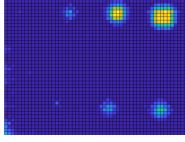
The simulation data analysis part of this thesis deepened the understanding of the Machine Learning (ML) algorithm implementation, regarding obtaining answers for the limits and robustness of the algorithm's damage detection capability. This was studied from the analysis of data obtained from the finite element analysis of lightweight plates modulated with different defect depth, and simulated with different mesh sizes. This robustness analysis involved the modelling of twenty four plate conditions (since each of the six defect depth plate conditions were simulated with four different mesh sizes), which was done by using Siemens Simcenter 3D<sup>®</sup> software. For each of the twenty four plate conditions (defect depth and

mesh), two versions of the ML algorithm were run. As conclusion remarks, some final considerations are presented regarding the mesh, the algorithm versions and the detection:

- The 8 mm mesh did not provide meaningful results (element size almost as equal as the smallest defect sizes);
- The 1 mm mesh provides no visual improvement comparing to the 2 mm mesh;
- The 2nd Gradient algorithm version provides results with better defect resolution than the mode shapes version;
- Defects can be visually detected up to 80% of thickness;
- Increasing the frequency band will provide the excitation of more Local Defect Resonance (LDR) modes, hence a better detection for higher Flat Bottom Hole (FBH) defect thickness.

### Match of all techniques

**Table 11.1:** Match of all results from the different algorithms for the experimental procedure

	Time-Domain			Frequency-Domain	
ML tool (K-means) CWT & STFT	2D CNN CWT & STFT	1D CNN	1D CNN	ML tool (GADAC) MS & 2nd Der	
					
					

During this thesis, for each of the presented algorithms on the experimental data analysis part, the results were presented on a single dataset measured on the PMMA plate. This was done in order to make a comparison between the algorithms, which is presented in table 11.1). Analysing these results and also remembering the discussion previously made for each algorithm, a number of conclusions can be taken regarding the performances:

- As shown in the discussion of results for the Machine Learning tool (K-means) in the time data procedure, the Short-Time Fourier Transform (STFT) data provided fewer outliers than the Continuous Wavelet Transform (CWT) data. Thus, although both algorithms detect 5 defects, there can be concluded the superiority of performance from the Short-Time Fourier Transform (STFT) version;
- The Machine Learning tool (GADAC ) mode shapes version provides more misclassifications and thus less clear images than the Machine Learning tool (GADAC ) 2nd Derivatives version, but has a bigger resolution in detecting the smaller defects.

- Regarding all the Machine Learning tools, there is a superiority of the ML tool (GADAC), since it detects 6 defects. Also, in terms of algorithm it is more complex both linking K-means and Multivariate Anomaly Detection, whereas the one for the time-domain contained only the K-means.
- Regarding both versions of the 2D CNN made to analyse either the CWT or the STFT images, both algorithms provide the detection of 6 defects plus the piezoelectric patch location (bottom left corner). In a comparative term, the CWT version outputs more misclassifications than the STFT version (similar to the case of the Machine Learning tool);
- Both 1D CNNs provide the detection of 7 defects plus the piezoelectric patch's location (bottom left corner). But looking at the frequency-domain one, there is a defect misclassification in the middle;
- The 1D CNN for the time-domain demonstrated to be the more robust damage classification algorithm, in a comparative overview of all algorithm's results for this dataset.

## Future developments

Considering the work done in this thesis project, in terms of algorithm development and their results, several developments can be made to improve their performances and several studies to better understand their variability and application to other cases. As future steps remarks, some suggestions are follow:

- Study the variance of these techniques (possible Monte Carlo simulations);
- Improve the clustering technique used in this project (K-means) to build unsupervised learning algorithms;
- Study the convolutional operations within the 1D CNN to better understand the feature extraction operations;
- Apply semantic segmentation techniques on the mode shape data with a damage detection purpose;
- Use other features obtained from the Continuous Wavelet Transform (CWT) and the Short-Time Fourier Transform (STFT) , such as the time of maximum vibration amplitude for the Machine Learning applications.



# Bibliography

- [1] R. P. Tavares, F. Otero, A. Turon, and P. P. Camanho, “Effective simulation of the mechanics of longitudinal tensile failure of unidirectional polymer composites,” *International Journal of Fracture*, vol. 208, no. 1-2, pp. 269–285, 2017.
- [2] R. P. Tavares, *Mechanics of deformation and failure of fibre hybrid composites*. PhD thesis, Faculdade de Engenharia da Universidade do Porto (FEUP), 2020.
- [3] S. Hall, “The effective management and use of structural health data,” in *Proceedings of the 2nd International Workshop on Structural Health Monitoring*, pp. 265–275, 1999.
- [4] S. S. Kessler, S. M. Spearing, M. J. Atalla, C. E. Cesnik, and C. Soutis, “Damage detection in composite materials using frequency response methods,” *Composites Part B: Engineering*, vol. 33, no. 1, pp. 87–95, 2002.
- [5] I. Solodov, M. Rahammer, and N. Gulnizkij, “Highly-sensitive and frequency-selective imaging of defects via local defect resonance,” in *Proceedings of European Conference on Non-destructive Testing (ECNDT 2014)*, vol. 6, 2014.
- [6] I. Solodov, “Resonant ultrasonic imaging of defects for advanced non-linear and thermosonic applications,” *International Journal of Microstructure and Materials Properties*, pp. 261–273, 2014.
- [7] J. Segers, S. Hedayatrasa, E. Verboven, G. Poelman, W. Van Paepegem, and M. Kersemans, “Efficient automated extraction of local defect resonance parameters in fiber reinforced polymers using data compression and iterative amplitude thresholding,” *Journal of Sound and Vibration*, vol. 463, p. 114958, 2019.
- [8] L. E. Mujica Delgado, J. Rodellar Benedé, and J. Vehí Casellas, “A review of impact damage detection in structures using strain data,” *International journal of COMADEM*, vol. 13, no. 1, pp. 3–18, 2010.
- [9] O. Janssens, V. Slavkovikj, B. Vervisch, K. Stockman, M. Loccufier, S. Verstockt, R. Van de Walle, and S. Van Hoecke, “Convolutional neural network based fault detection for rotating machinery,” *Journal of Sound and Vibration*, vol. 377, pp. 331–345, 2016.

- [10] O. Abdeljaber, O. Avci, S. Kiranyaz, M. Gabbouj, and D. J. Inman, "Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks," *Journal of Sound and Vibration*, vol. 388, pp. 154–170, 2017.
- [11] J. C. Aldrin and D. S. Forsyth, "Demonstration of using signal feature extraction and deep learning neural networks with ultrasonic data for detecting challenging discontinuities in composite panels," in *AIP Conference Proceedings*, vol. 2102, p. 020012, AIP Publishing, 2019.
- [12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.
- [13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.
- [15] O. Avci, O. Abdeljaber, S. Kiranyaz, and D. Inman, "Structural damage detection in real time: implementation of 1d convolutional neural networks for shm applications," in *Structural Health Monitoring & Damage Detection*, vol. 7, pp. 49–54, Springer, 2017.
- [16] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, "Real-time motor fault detection by 1-d convolutional neural networks," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 11, pp. 7067–7075, 2016.
- [17] W. Zhang, C. Li, G. Peng, Y. Chen, and Z. Zhang, "A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load," *Mechanical Systems and Signal Processing*, vol. 100, pp. 439–453, 2018.
- [18] F. Jia, Y. Lei, J. Lin, X. Zhou, and N. Lu, "Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data," *Mechanical Systems and Signal Processing*, vol. 72, pp. 303–315, 2016.
- [19] B. Lopes, C. Colangeli, K. Janssens, A. Mroz, and H. Van der Auweraer, "Neural network models for the subjective and objective assessment of a propeller aircraft interior sound quality," in *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, vol. 259, pp. 4124–4135, Institute of Noise Control Engineering, 2019.
- [20] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ecg classification by 1-d convolutional neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 664–675, 2015.
- [21] M. de Oliveira, A. Monteiro, and J. Vieira Filho, "A new structural health monitoring strategy based on pzt sensors and convolutional neural network," *Sensors*, vol. 18, no. 9, p. 2955, 2018.

- [22] A. Shihavuddin, X. Chen, V. Fedorov, A. Nymark Christensen, N. Andre Brogaard Riis, K. Branner, A. BJORHOLM DAHL, and R. Reinhold Paulsen, "Wind turbine surface damage detection by deep learning aided drone inspection analysis," *Energies*, vol. 12, no. 4, p. 676, 2019.
- [23] C. V. Dung *et al.*, "Autonomous concrete crack detection using deep fully convolutional neural network," *Automation in Construction*, vol. 99, pp. 52–58, 2019.
- [24] O. Wyman, "Global fleet & mro market forecast commentary," *Electronic Article*. url: <https://www.oliverwyman.com/content/dam/oliver-wyman/v2/publications/2019/January/global-fleet-mro-market-forecast-commentary-2019-2029.pdf> (visited on 18/05/2020), 2019.
- [25] IATA, "Airline maintenance cost executive commentary," *An exclusive benchmark analysis (FY2018 data) by IATA's maintenance cost technical group*, 2018.
- [26] B. Peeters, W. Hendricx, J. Debille, and H. Climent, "Modern solutions for ground vibration testing of large aircraft," *Sound and vibration*, vol. 43, no. 1, p. 8, 2009.
- [27] I. Solodov, J. Bai, S. Bekgulyan, and G. Busse, "A local defect resonance to enhance acoustic wave-defect interaction in ultrasonic nondestructive evaluation," *Applied Physics Letters*, vol. 99, no. 21, p. 211911, 2011.
- [28] J. Segers, S. Hedayatrasa, E. Verboven, G. Poelman, W. Van Paepegem, and M. Kersemans, "In-plane local defect resonances for efficient vibrothermography of impacted carbon fiber-reinforced polymers (cfrp)," *NDT & E International*, vol. 102, pp. 218–225, 2019.
- [29] T. M. Mitchell *et al.*, "Machine learning. 1997," *Burr Ridge, IL: McGraw Hill*, vol. 45, no. 37, pp. 870–877, 1997.
- [30] Y. Le Cun, L. D. Jackel, B. Boser, J. S. Denker, H. P. Graf, I. Guyon, D. Henderson, R. E. Howard, and W. Hubbard, "Handwritten digit recognition: Applications of neural network chips and automatic learning," *IEEE Communications Magazine*, vol. 27, no. 11, pp. 41–46, 1989.
- [31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>, accessed on December, 2019.
- [32] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, pp. 2951–2959, 2012.
- [33] D. R. Jones, "A taxonomy of global optimization methods based on response surfaces," *Journal of global optimization*, vol. 21, no. 4, pp. 345–383, 2001.
- [34] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

- [35] Y. Zhang, K. Sohn, R. Villegas, G. Pan, and H. Lee, "Improving object detection with deep convolutional networks via bayesian optimization and structured prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 249–258, 2015.
- [36] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.
- [37] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.
- [38] B. Peeters, H. Van der Auweraer, P. Guillaume, and J. Leuridan, "The polymax frequency-domain method: a new standard for modal parameter estimation?," *Shock and Vibration*, vol. 11, no. 3, 4, pp. 395–409, 2004.
- [39] J. Segers, S. Hedayatrasa, G. Poelman, W. Van Paepegem, and M. Kersemans, "Probing the limits of full-field linear local defect resonance identification for deep defect detection," *Ultrasonics*, p. 106130, 2020.
- [40] E. Di Lorenzo, Q. L. Bernabei, and B. Peeters, "Non-destructive testing for damage detection in composite materials," *Structural Health Monitoring 2019*, 2019.



# Appendix A

## 1D CNN architectures

### A.1 First 1D CNN

Table A.1 presents the architecture used for both the frequency-domain and time-domain 1D Convolutional Neural Networks (CNNs), which are presented in section 7.4 and section 8.3.

**Table A.1:** 1D CNN architecture used for the time and frequency-domain analysis

Block	Layer	Layer Type	Kernel Size	Kernel No.	Stride	Padding
<b>B1</b>	1	Conv	61x1	16	1	Yes
	2	BN	-	-	-	-
	3	Max Pool	8x1	16	2	No
<b>B2</b>	4	Conv	3x1	32	1	Yes
	5	BN	-	-	-	-
	6	Max Pool	2x1	32	2	No
<b>B3</b>	7	Conv	3x1	64	1	Yes
	8	BN	-	-	-	-
	9	Max Pool	2x1	64	2	No
<b>B4</b>	10	Conv	3x1	64	1	Yes
	11	BN	-	-	-	-
	12	Max Pool	2x1	64	2	No
<b>B5</b>	13	Conv	3x1	64	1	Yes
	14	BN	-	-	-	-
	15	Dropout	-	-	-	-
<b>B6</b>	16	FC	200	-	-	-
	17	FC	2	-	-	-
	18	Softmax	-	-	-	-

## A.2 1D CNNs obtained with the Bayesian Optimization

### A.2.1 For Time-domain data

Table A.2 presents the architecture obtained with the Bayesian Optimization for the time-domain 1D CNN, discussed in sub-section 7.4.3.

**Table A.2:** Optimized architecture for the time-domain 1D CNN

Block	Layer	Layer Type	Kernel Size	Kernel No.	Stride	Padding
<b>B1</b>	1	Conv	61x1	16	1	Yes
	2	BN	-	-	-	-
	3	Max Pool	8x1	16	2	No
<b>B2</b>	4	Conv	3x1	32	1	Yes
	5	BN	-	-	-	-
	6	Max Pool	2x1	32	2	No
<b>B3</b>	7	Conv	3x1	64	1	Yes
	8	BN	-	-	-	-
	9	Max Pool	2x1	64	2	No
<b>B4</b>	10	Drop 40%	-	-	-	-
	11	FC	200	-	-	-
	12	Drop 40%	-	-	-	-
	13	FC	100	-	-	-
	14	FC	2	-	-	-
	15	Softmax	-	-	-	-

## A.2.2 For Frequency-domain data

Table A.3 presents the architecture obtained with the Bayesian Optimization for the frequency-domain 1D CNN, discussed in sub-section 8.3.3.

**Table A.3:** Optimized architecture for the frequency-domain 1D CNN

Block	Layer	Layer Type	Kernel Size	Kernel No.	Stride	Padding
<b>B1</b>	1	Conv	61x1	16	1	Yes
	2	BN	-	-	-	-
	3	Max Pool	8x1	16	2	No
<b>B2</b>	4	Conv	3x1	32	1	Yes
	5	BN	-	-	-	-
	6	Max Pool	2x1	32	2	No
<b>B3</b>	7	Conv	3x1	64	1	Yes
	8	BN	-	-	-	-
	9	Max Pool	2x1	64	2	No
<b>B4</b>	10	Drop 30%	-	-	-	-
	11	FC	200	-	-	-
	12	Drop 30%	-	-	-	-
	13	FC	100	-	-	-
	14	FC	2	-	-	-
	15	Softmax	-	-	-	-

## Appendix B

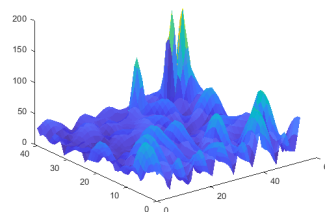
# Machine Learning tools applied to singular mode shapes and 2nd Gradients

### B.1 Experimental mode shapes

The ML tool (GADAC) described in section 8.2 analyses sequentially the mode shapes or its 2nd Gradients (Laplacian, Gaussian Curvature and 2nd Derivatives). The way the algorithm works is displayed in figures 8.8 and 8.10, where the final result is the sum of the singular results for all mode shapes. In this appendix section, the results from the analysis of one mode shape or a consequent 2nd Gradient are shown. Figure B.1b shows a baseline mode shape which will be analysed directly by the ML tool (GADAC), or submit to the three studied versions of performing 2nd Gradient techniques and after to the ML tool (GADAC).



(a) Original PMMA plate bottom surface view.



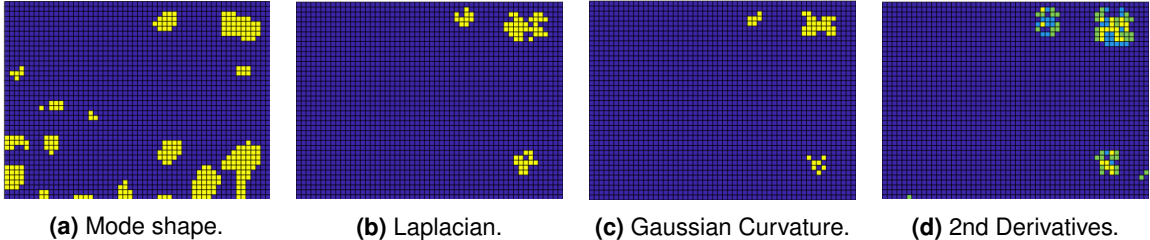
(b) Base mode shape to be input in the ML tool (GADAC).

**Figure B.1:** PMMA plate and corresponding high-frequency mode shape.

The classification performed by this algorithm is binary, hence all points of the mode shape, correspondent to points measured on top of the plate, will be classified in either defected (yellow) or non-defected (blue). Figure B.2 shows the results obtained through this classification for each version of

the input data. The analysis of the algorithm for each of the 2nd Gradient techniques (Figure B.2 b - d)) accurately detects three of the plate's damages, the two bigger squared and the bigger circular damages (Figure B.1a). This is coherent with the aim of applying these techniques in order to highlight the presence of a defect in a mode shape, making it easier to detect. The difference between the results of the 2nd Derivatives version in comparison to the Laplacian and Gaussian Curvatures versions lies in the fact that the algorithm is applied independently to each of the 2nd Derivatives ( $\frac{d^2}{dx^2}$ ;  $\frac{d^2}{dy^2}$ ;  $\frac{d^2}{dxdy}$ ) and the result is the sum of those three outputs.

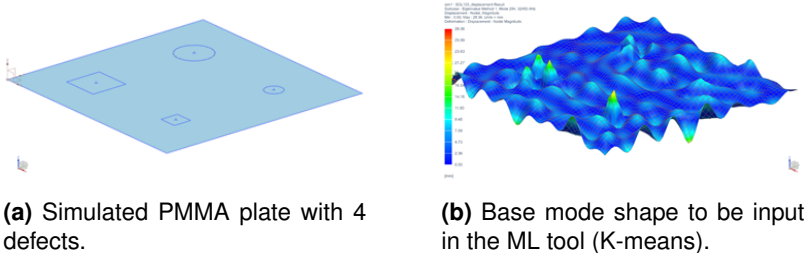
The analysis of the mode shape, shown in figure B.2 a), results in many misclassifications across the plate. However, it was found that, although the results for a singular mode shape seem to be less accurate, the overall result obtained through the analysis of all mode shapes in the 40 kHz frequency band is accurate in classifying the defects. Moreover, comparing to the 2nd Gradient versions, it has increased accuracy in detecting the smallest defects, although the overall result provides more misclassifications.



**Figure B.2:** Results of the ML tool (GADAC) applied on a single mode shape (figure B.1b) and its 2nd Gradients

## B.2 Simulated mode shapes

Similar to the experimental procedure of analysing mode shapes, a ML tool (K-means) was developed to analyse mode shapes obtained through simulation, which is presented in section 10.2. This algorithm also analyses all mode shapes obtained in the large frequency band, but in this section, the results obtained for the analysis of one mode shape or the corresponding 2nd Gradient will be presented. Figure B.3b shows a baseline mode shape which will be analysed directly by the ML tool (K-means), or submit to the three versions of performing 2nd Gradient techniques (Laplacian, Gaussian Curvature, 2nd Derivatives) and after to the ML tool (K-means).

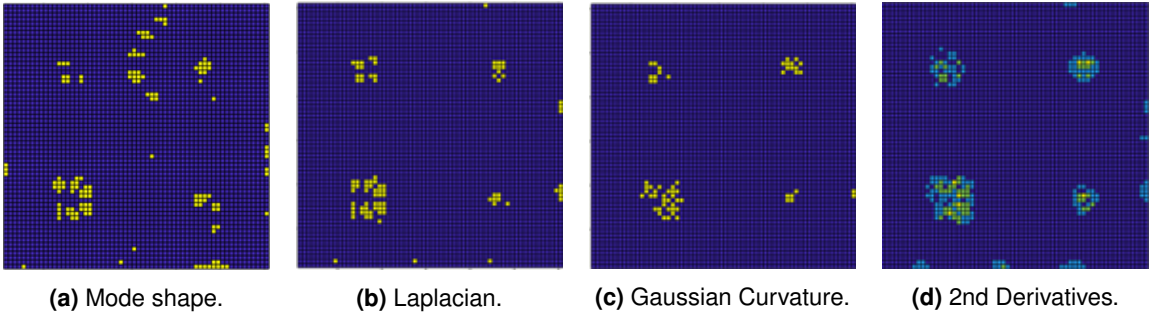


**Figure B.3:** Simulated plate and corresponding high-frequency mode shape.

The results obtained for each version of the input data are presented in figure B.4, where each of the

sub-figures has the result of the algorithm's binary classification. Similar conclusions to the ones taken in section B.1 regarding the experimental data, can be taken for the simulation mode shapes. The 2nd Gradient results (figures B.4 b) - d)) have less misclassifications than the mode shape results, due to the fact that these techniques highlight the behaviour of a damage in a mode shape. As can be seen, many points are classified as damaged (yellow points) in the locations of the four damages from the simulated plate, in figure B.3a.

In a close analysis of these results, some misclassifications in the lines closer to the borders of the plate can be seen for the results of the 2nd Gradient versions. This can be associated with the application of the 2nd Gradients to the borders of the plate, which implies a loss of correlation.



**Figure B.4:** Results of the ML tool (K-means) applied on a single mode shape (figure 8.5a) and its 2nd Gradients